7.5

IBM Message Service Client for .NET

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información en “Avisos” en la página 255.

Esta edición se aplica a la versión 7 release 5 de IBM® WebSphere MQ y a todos los releases y modificaciones posteriores hasta que se indique lo contrario en nuevas ediciones.

Cuando envía información a IBM, otorga a IBM un derecho no exclusivo para utilizar o distribuir la información de la forma que considere adecuada, sin incurrir por ello en ninguna obligación con el remitente.

Contenido

Introducción a Message Service Client for .NET

Message Service Client for .NET proporciona una interfaz de programación de aplicaciones (API) denominada XMS que tiene el mismo conjunto de interfaces que Java Message Service (JMS) API. Message Service Client for .NET contiene una implementación totalmente gestionada de XMS, que puede utilizar cualquier lenguaje compatible con .NET.

XMS admite:

- Mensajería de punto a punto
- Mensajería de Publicar/Suscribir
- Entrega de mensajes síncrona
- Entrega de mensajes asíncrona

La aplicación Un XMS puede intercambiar mensajes con los siguientes tipos de aplicación:

- Aplicación Un XMS
- Una aplicación IBM WebSphere MQ classes for JMS
- Una aplicación IBM WebSphere MQ nativa
- Una aplicación JMS que utiliza el proveedor de mensajería predeterminado de WebSphere

Una aplicación XMS se puede conectar a, y utilizar los recursos de, cualquiera de los servidores de mensajería siguientes:

IBM WebSphere MQ gestor de colas
La aplicación se puede conectar en la modalidad de enlaces o cliente.

WebSphere Application Server Bus de integración de Integration Bus
La aplicación puede utilizar una conexión TCP/IP directa, o puede utilizar HTTP sobre TCP/IP.

WebSphere Event Broker o WebSphere Message Broker
Los mensajes se transportan entre la aplicación y el intermediario utilizando WebSphere MQ Real-Time Transport. Los mensajes se pueden entregar a la aplicación utilizando WebSphere MQ Multicast Transport.

Al conectarse a un gestor de colas de IBM WebSphere MQ , la aplicación Un XMS puede utilizar IBM WebSphere MQ Enterprise Transport para comunicarse con WebSphere Event Broker o WebSphere Message Broker. De forma alternativa, la aplicación Un XMS puede publicar y suscribirse conectándose a IBM WebSphere MQ.

Conceptos relacionados

“Estilos de mensajería” en la página 7

“El modelo de objeto XMS” en la página 7
La API XMS es una interfaz orientada a objetos. El modelo de objeto de XMS está basado en el modelo de objeto de JMS 1.1.

“El modelo de mensaje XMS” en la página 10
El modelo de mensaje XMS es el mismo que el modelo de mensaje IBM WebSphere MQ classes for JMS.

Introducción a Message Service Client for .NET

Message Service Client for .NET proporciona una interfaz de programación de aplicaciones (API) denominada XMS que tiene el mismo conjunto de interfaces que Java Message Service (JMS) API. Message Service Client for .NET contiene una implementación totalmente gestionada de XMS, que puede utilizar cualquier lenguaje compatible con .NET.

XMS admite:

- Mensajería de punto a punto

- Mensajería de Publicar/Suscribir
- Entrega de mensajes síncrona
- Entrega de mensajes asíncrona

La aplicación Un XMS puede intercambiar mensajes con los siguientes tipos de aplicación:

- Aplicación Un XMS
- Una aplicación IBM WebSphere MQ classes for JMS
- Una aplicación IBM WebSphere MQ nativa
- Una aplicación JMS que utiliza el proveedor de mensajería predeterminado de WebSphere

Una aplicación XMS se puede conectar a, y utilizar los recursos de, cualquiera de los servidores de mensajería siguientes:

IBM WebSphere MQ gestor de colas
La aplicación se puede conectar en la modalidad de enlaces o cliente.

WebSphere Application Server Bus de integración de Integration Bus
La aplicación puede utilizar una conexión TCP/IP directa, o puede utilizar HTTP sobre TCP/IP.

WebSphere Event Broker o WebSphere Message Broker
Los mensajes se transportan entre la aplicación y el intermediario utilizando WebSphere MQ Real-Time Transport. Los mensajes se pueden entregar a la aplicación utilizando WebSphere MQ Multicast Transport.

Al conectarse a un gestor de colas de IBM WebSphere MQ , la aplicación Un XMS puede utilizar IBM WebSphere MQ Enterprise Transport para comunicarse con WebSphere Event Broker o WebSphere Message Broker. De forma alternativa, la aplicación Un XMS puede publicar y suscribirse conectándose a IBM WebSphere MQ.

Conceptos relacionados

"Estilos de mensajería" en la página 7

"El modelo de objeto XMS" en la página 7
La API XMS es una interfaz orientada a objetos. El modelo de objeto de XMS está basado en el modelo de objeto de JMS 1.1.

"El modelo de mensaje XMS" en la página 10
El modelo de mensaje XMS es el mismo que el modelo de mensaje IBM WebSphere MQ classes for JMS.

Novedades de este release

Hay una serie de mejoras en este release de Message Service Client for .NET.

"Lectura y escritura del descriptor de mensaje desde una aplicación Message Service Client for .NET" en la página 85
Puede acceder a todos los campos de descriptor de mensaje de un mensaje IBM WebSphere MQ excepto `StrucId` y `Version`. El campo `BackoutCount` se puede leer pero no se puede escribir en él. El acceso a los campos sólo está disponible cuando se conecta a un gestor de colas de IBM WebSphere MQ versión 6 y superior. El acceso se controla mediante las propiedades de destino descritas más adelante.

"Las aplicaciones de ejemplo" en la página 19
Las aplicaciones de ejemplo XMS proporcionan una visión general de las características comunes de cada API. Puede utilizarlos para verificar la instalación y la configuración del servidor de mensajería, y para verificar sus propias aplicaciones.

Mejoras de rendimiento
Se ha mejorado el rendimiento de XMS .NET.

"Reconexión automática del cliente IBM WebSphere MQ a través de XMS" en la página 46
Puede configurar un cliente WebSphere MQ V7.1 XMS IBM WebSphere MQ para que se vuelva a conectar automáticamente después de una anomalía de red, gestor de colas o servidor.

“Transacciones XA IBM WebSphere MQ gestionadas a través de XMS” en la página 41
Las transacciones XA de WebSphere MQ gestionadas se pueden utilizar a través de XMS.

GMO_CONVERT
La especificación de un valor `GMO_CONVERT` en un mensaje es opcional. Si se especifica un valor `GMO_CONVERT`, la conversión se realiza de acuerdo con el valor especificado.

Estilos de mensajería

XMS admite los estilos de mensajería punto a punto y Publicar/Suscribir.

Los estilos de mensajería también se denominan dominios de mensajería.

Mensajería de punto a punto

Una forma común de mensajería de punto a punto utiliza la colocación en colas. En el caso más sencillo, una aplicación envía un mensaje a otra aplicación identificando, de forma implícita o explícita, una cola de destino. La sistema subyacente de mensajería y colocación en colas recibe el mensaje de la aplicación emisora y direcciona el mensaje a su cola de destino. La aplicación receptora puede recuperar el mensaje de la cola.

Si el sistema subyacente de mensajería y colocación en cola contiene WebSphere Message Broker, WebSphere Message Broker podría duplicar un mensaje y direccionar copias del mensaje a distintas colas. Como resultado, más de una aplicación puede recibir el mensaje. WebSphere Message Broker también podría transformar un mensaje y añadirle datos.

Una característica clave de la mensajería punto a punto es que una aplicación coloca un mensaje en una cola local cuando envía un mensaje. EL sistema subyacente de mensajería y colocación en cola determina a qué cola de destino se envía el mensaje. La aplicación receptora recupera el mensaje de la cola de destino.

Mensajería de Publicación/suscripción

En la mensajería Publicar/Suscribir, hay dos tipos de aplicación: publicador y suscriptor.

Un *publicador* proporciona información en forma de mensajes de publicación. Cuando un publicador publica un mensaje, especifica un tema, que identifica el tema de la información incluida en el mensaje.

Un *suscriptor* es un consumidor de la información que se publica. Un suscriptor especifica los temas en los que está interesado creando suscripciones.

El sistema de publicación/suscripción recibe publicaciones de publicadores y suscripciones de suscriptores. Direcciona publicaciones a suscriptores. Un suscriptor recibe publicaciones sobre solo aquellos temas a los que está suscrito.

Una característica clave de la mensajería Publicar/Suscribir es que un publicador identifica un tema cuando publica un mensaje. No identifica los suscriptores. Si se publica un mensaje sobre un tema para el cual no hay suscriptores, ninguna aplicación recibe el mensaje.

Una aplicación puede ser a la vez publicador y suscriptor.

El modelo de objeto XMS

La API XMS es una interfaz orientada a objetos. El modelo de objeto de XMS está basado en el modelo de objeto de JMS 1.1.

La lista siguiente resume las clases principales de XMS, o tipos de objeto:

`ConnectionFactory`
Un objeto `ConnectionFactory` encapsula un conjunto de parámetros para una conexión. Una aplicación utiliza un `ConnectionFactory` para crear una conexión. Una aplicación puede proporcionar los parámetros durante el tiempo de ejecución y crear un objeto `ConnectionFactory`. De forma alternativa, los parámetros de conexión se pueden almacenar en un repositorio de

objetos administrados. Una aplicación puede recuperar un objeto del repositorio y crear un objeto `ConnectionFactory` a partir del mismo.

`Connection`
Un objeto `Connection` encapsula una conexión activa de una aplicación a un servidor de mensajería. Una aplicación utiliza una conexión para crear sesiones.

`Destination`
Una aplicación envía mensajes o recibe mensajes utilizando un objeto `Destination`. En el dominio Publicar/Suscribir, un objeto `Destination` encapsula un tema y, en el dominio punto a punto, un objeto `Destination` encapsula una cola. Una aplicación puede proporcionar los parámetros para crear un objeto `Destination` durante el tiempo de ejecución. De forma alternativa, puede crear un objeto `Destination` a partir de una definición de objeto que se almacena en el repositorio de objetos administrados.

`Session`
Un objeto `Session` es un contexto de hebra única para enviar y recibir mensajes. Una aplicación utiliza un objeto `Session` para crear objetos `Message`, `MessageProducer` y `MessageConsumer`.

`Message`
Un objeto `Message` encapsula el objeto `Message` que envía una aplicación utilizando un objeto `MessageProducer` o que recibe utilizando un objeto `MessageConsumer`.

`MessageProducer`
Una aplicación utiliza un objeto `MessageProducer` para enviar mensajes a un destino.

`MessageConsumer`
Una aplicación utiliza un objeto `MessageConsumer` para recibir mensajes enviados a un destino.

Figura 1 en la página 8 muestras estos objetos y sus relaciones.

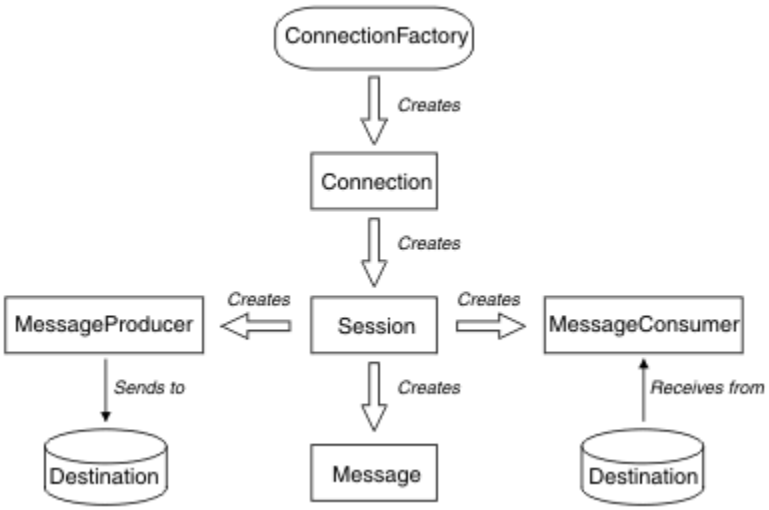

Figura 1. Objetos XMS y sus relaciones

En .NET, las clases XMS se definen como un conjunto de interfaces .NET. Al codificar aplicaciones XMS .NET, solo necesitará las interfaces declaradas.

El modelo de objeto de XMS está basado en las interfaces independientes del dominio que se describen en *Java Message Service Specification, Versión 1.1*. Las clases específicas de dominio como, por ejemplo, `Topic`, `TopicPublisher` y `TopicSubscriber`, no se proporcionan.

Atributos y propiedades de objetos

Un objeto XMS puede tener atributos y propiedades, que son característicos del objeto, que se implementan de formas diferentes.

Atributos

Una característica de objeto que siempre está presente y ocupa almacenamiento, aunque el atributo no tenga un valor. En este sentido, un atributo es similar a un campo en una estructura de datos de longitud fija. Una característica distintiva de atributos es que cada atributo tiene sus propios métodospara establecer y obtener su valor.

Propiedades

Una propiedad de un objeto está presente y ocupa almacenamiento solo después de que su valor se haya establecido. Una propiedad no se puede suprimir o su almacenamiento se ha recuperado después de que su valor se haya establecido. Puede cambiar su valor. XMS proporciona un conjunto de métodos genéricos para establecer y obtener valores de propiedad.

Conceptos relacionados

Tipos primitivos de XMS

XMS proporciona equivalentes de los ochos tipos primitivos de Java (byte, short, int, long, float, double, char y boolean). Esto permite el intercambio de mensajes entre XMS y JMS sin pérdida ni corrupción de datos.

Conversión implícita de un valor de propiedad de un tipo de datos a otro

Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación

Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Objetos administrados

Mediante el uso de objetos administrados, puede administrar los valores de la conexión utilizada por aplicaciones cliente que se van a administrar desde un repositorio central. Una aplicación recupera definiciones de objeto del repositorio central y las utiliza para crear objetos `ConnectionFactory` y `Destination`. Utilizando objetos administrados, puede desacoplar aplicaciones de los recursos que utilizan durante el tiempo de ejecución.

Por ejemplo, las aplicaciones XMS se pueden escribir y probar con objetos administrados que hacen referencia a un conjunto de conexiones y destinos en un entorno de prueba. Cuando se despliegan las aplicaciones, los objetos administrados se pueden cambiar para configurar las aplicaciones para hacer referencia a conexiones y destinos en el entorno de producción.

XMS admite dos tipos de objeto administrado:

- Un objeto `ConnectionFactory`, que es utilizado por aplicaciones para realizar la conexión inicial al servidor.
- Un objeto `Destination`, que es utilizado por aplicaciones para especificar el destino para mensajes que se están enviando, y el origen de mensajes que se están recibiendo. Un destino es un tema o una cola del servidor al que se conecta una aplicación.

La herramienta de administración **JMSAdmin** se proporciona con IBM WebSphere MQ. Se utiliza para crear y gestionar objetos administrados para en un repositorio central de objetos administrados.

Los objetos administrados del repositorio pueden ser utilizados por aplicaciones IBM WebSphere MQ classes for JMS y XMS. Las aplicaciones XMS pueden utilizar los objetos `ConnectionFactory` y `Destination` para conectarse a un IBM WebSphere MQ gestor de colas. Un administrador puede cambiar las definiciones de objeto incluidas en el repositorio sin que haya repercusiones en el código de aplicación.

El diagrama siguiente muestra cómo una aplicación XMS suele utilizar los objetos administrados.

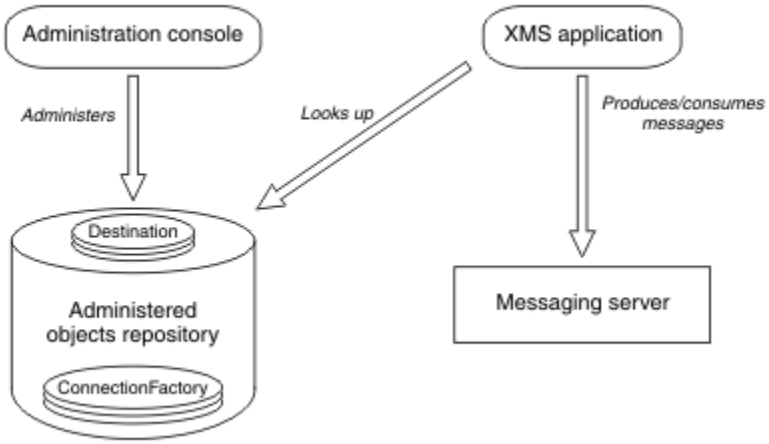

Figura 2. Uso típico de objetos administrados por para de una aplicación XMS

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

El modelo de mensaje XMS

El modelo de mensaje XMS es el mismo que el modelo de mensaje IBM WebSphere MQ classes for JMS.

En particular, XMS implementa los mismos campos de cabecera de mensaje y las mismas propiedades de mensaje que implementa IBM WebSphere MQ classes for JMS:

- Campos de cabecera JMS. Estos campos tienen nombres que empiezan con el prefijo JMS.
- Propiedades definidas por JMS. Estos campos tienen propiedades cuyos nombres empiezan con el prefijo JMSX.
- Propiedades definidas de IBM . Estos campos tienen propiedades cuyos nombres empiezan con el prefijo JMS_IBM_.

Como resultado, las aplicaciones XMS pueden intercambiar mensajes con aplicaciones IBM WebSphere MQ classes for JMS. En cada mensaje, algunos de los campos y de las propiedades de cabecera están establecidos por la aplicación y otros están establecidos por XMS o IBM WebSphere MQ classes for JMS. Algunos de los campos establecidos por XMS o IBM WebSphere MQ classes for JMS se establecen cuando se envía el mensaje, y otros cuando se recibe. Los campos y las propiedades de cabecera se propagan con un mensaje a través de un servidor de mensajería, cuando proceda. Pasan a estar disponibles para cualquier aplicación que recibe el mensaje.

Impedir que las aplicaciones utilicen una versión XMS más nueva

De forma predeterminada, cuando se instala una versión de XMS más nueva, las aplicaciones que utilizan la versión anterior cambian automáticamente a la versión más nueva sin tener que recompilar.

Acerca de esta tarea

La característica de coexistencia de varias versiones garantiza que la instalación de una versión de XMS más nueva no sobrescribe la versión anterior de XMS. En lugar de esto, coexisten varias instancias de conjuntos similares de XMS .NET en la Global Assembly Cache (GAC), pero tienen números de versión diferentes. De forma interna, la GAC utiliza un archivo de política para direccionar las llamadas de aplicación a la última versión de XMS. Las aplicaciones se ejecutan sin necesidad de recompilar y pueden utilizar nuevas características disponibles en la versión más nueva de XMS .NET.

Sin embargo, si una aplicación es necesaria para utilizar la versión anterior de XMS , establezca el atributo `publisherpolicy` en no en el archivo de configuración de la aplicación.

Nota: Un archivo de configuración de aplicación es un archivo con un nombre que está formado por el nombre del programa ejecutable con el que está relacionado el archivo, con el sufijo .config. Por ejemplo, el archivo de configuración de aplicación para text.exe tendría el nombre text.exe.config.

Sin embargo, en cualquier momento, todas las aplicaciones de un sistema utilizan la misma versión de XMS .NET.

Configuración del entorno de servidor de mensajería

Este secciónCapítulo describe cómo configurar el entorno del servidor de mensajería para permitir que las aplicaciones XMS se conecten a un servidor.

Para las aplicaciones que se conectan a un gestor de colas de IBM WebSphere MQ, es necesario el cliente IBM WebSphere MQ (o gestor de colas para la modalidad de enlaces).

Actualmente no hay ningún requisito previo para las aplicaciones que utilizan una conexión en tiempo real con un intermediario.

Debe configurar el entorno de servidor de mensajería antes de ejecutar cualquier aplicación XMS, incluyendo las aplicaciones de ejemplo proporcionadas con XMS.

Este secciónCapítulo contiene los temassecciones siguientes:

- "Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ" en la página 11
- "Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario" en la página 13
- "Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus" en la página 14

Tareas relacionadas

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ

En esta sección se presupone que está utilizando IBM WebSphere MQ versión 7.0. Para poder ejecutar una aplicación que se conecta a un IBM WebSphere MQ gestor de colas, debe configurar el gestor de colas. Para una aplicación de Publicar/Suscribir, es necesaria alguna configuración adicional si está utilizando la interfaz de publicación/suscripción en cola.

Antes de empezar

XMS funciona con WebSphere Message Broker versión 6.1 o anterior.

Antes de iniciar esta tarea, realice los pasos siguientes:

- Asegúrese de que la aplicación tiene acceso a un gestor de colas que esté en ejecución.

- Si la aplicación es una aplicación de publicación/suscripción y utiliza la interfaz de publicación/suscripción en cola, asegúrese de que el atributo "PSMODE" está establecido en "ENABLED" en el gestor de colas.
- Asegúrese de que la aplicación utiliza una fábrica de conexiones cuyas propiedades se han establecido correctamente para conectarse al gestor de colas. Si la aplicación es una aplicación de publicación/suscripción, asegúrese de que las propiedades de fábrica de conexiones apropiadas se establecen para utilizar el intermediario. Si desea más información sobre las propiedades de una fábrica de conexiones, consulte “Propiedades de ConnectionFactory” en la página 185.

Acerca de esta tarea

Configure el gestor de colas y el intermediario para ejecutar aplicaciones XMS de la misma manera que configura el gestor de colas y la interfaz de publicación/suscripción en cola para ejecutar aplicaciones JMS de WebSphere MQ. Los pasos siguientes resumen lo qué debe hacer.

Procedimiento

1. En el gestor de colas, cree las colas que necesita la aplicación.

   Para obtener información sobre cómo crear colas, consulte el tema *Definición de colas* en la documentación del producto *WebSphere MQ*.

   Si la aplicación es una aplicación de Publicar/Suscribir y utiliza la interfaz de publicación/suscripción en cola que necesita acceder a colas del sistema IBM WebSphere MQ classes for JMS, espere hasta el Paso 4a antes de crear las colas.
2. Otorgue al ID de usuario asociado a la aplicación autorización para conectar con el gestor de colas, y la autorización necesaria para acceder a las colas.

   Para obtener información sobre la autorización, consulte la sección *Seguridad* de la *documentación del productoIBM WebSphere MQ*. Si la aplicación se conecta a gestor de colas en modalidad de cliente, consulte también el tema *Clientes* en la *documentación del productoIBM WebSphere MQ*.
3. Si la aplicación conecta con el gestor de colas en la modalidad de cliente, asegúrese de que esté definido un canal de conexión de servidor en el gestor de colas y que haya proceso de escucha iniciado.

   Para obtener información sobre cómo hacerlo, consulte el tema *Clientes* en la *documentación del productoIBM WebSphere MQ*.

   No es necesario que ejecute este paso para cada aplicación que conecte con el gestor de colas. Una definición de canal de conexión de servidor y un escucha pueden dar soporte a todas las aplicaciones que se conectan en la modalidad de cliente.
4. Si la aplicación es una aplicación de Publicar/Suscribir y utiliza la interfaz de publicación/suscripción en cola, siga los pasos siguientes.
   a) En el gestor de colas, cree las colas del sistema IBM WebSphere MQ classes for JMS ejecutando el script de los mandatos MQSC que se proporcionan con IBM WebSphere MQ. Asegúrese de que el ID de usuario asociado a WebSphere Message Broker tenga autorización para acceder a las colas.

      Para obtener información sobre dónde encontrar el script y cómo ejecutarlo, consulte el tema *Utilización de Java™* en la documentación del producto *WebSphere MQ*.

      Ejecute es paso una sola vez para el gestor de colas. Un mismo conjunto de colas del sistema IBM WebSphere MQ classes for JMS puede dar soporte a todas las aplicaciones XMS y IBM WebSphere MQ classes for JMS que conectan con el gestor de colas.

   b) Otorgue al ID de usuario asociado a la aplicación la autoridad para acceder a las colas del sistema IBM WebSphere MQ classes for JMS.

      Para obtener información sobre las autorizaciones que necesita el ID de usuario, consulte el tema *Utilización de Java* en la documentación del producto *IBM WebSphere MQ*.

c) Para un intermediario de WebSphere Event Broker o WebSphere Message Broker, cree y despliegue un flujo de mensajes para prestar servicio a la cola donde las aplicaciones envían mensajes que publican.

   El flujo de mensajes básico engloba un nodo de proceso de mensajes MQInput para leer los mensajes publicados y un nodo de proceso de mensajes Publication para publicar los mensajes.

   Para obtener información sobre cómo crear y desplegar un flujo de mensajes, consulte la documentación del producto WebSphere Event Broker o WebSphere Message Broker .

   No es necesario que realice este paso si ya se ha desplegado un flujo de mensajes apropiado en el intermediario.

Resultados

Ahora puede iniciar la aplicación.

Tareas relacionadas

Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario
Antes de poder ejecutar una aplicación que utiliza una conexión en tiempo real con un intermediario, debe configurar ese intermediario.

Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus
Antes de poder ejecutar una aplicación que se conecta a un WebSphere Servicio Integration Bus, debe configurar el bus de integración de servicios del mismo modo que configura el bus de integración de servicios para ejecutar aplicaciones JMS que utilizan el proveedor de mensajería predeterminado.

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Referencia relacionada

Requisitos previos para las aplicaciones XMS que se conectan a IBM WebSphere MQ
Algunos requisitos previos se aplican si la aplicación XMS se conecta a IBM WebSphere MQ.

Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario

Antes de poder ejecutar una aplicación que utiliza una conexión en tiempo real con un intermediario, debe configurar ese intermediario.

Antes de empezar

XMS funciona con WebSphere Message Broker versión 6.1 o anterior.

Antes de iniciar esta tarea, realice los pasos siguientes:

- Asegúrese de que la aplicación tiene acceso a un intermediario que se está ejecutando.
- Asegúrese de que la aplicación utiliza una fábrica de conexiones cuyas propiedades se han establecido de forma adecuada para una conexión en tiempo real con un intermediario. Si desea más información sobre las propiedades de una fábrica de conexiones, consulte “Propiedades de ConnectionFactory” en la página 185.

Acerca de esta tarea

Configure un intermediario para ejecutar aplicaciones XMS de la misma forma que se configura un intermediario para ejecutar aplicaciones IBM WebSphere MQ classes for JMS. Los pasos siguientes resumen lo que debe hacer pero, para obtener más detalles, consulte la documentación del producto WebSphere Event Brokero WebSphere Message Broker :

Procedimiento

1. Crear y desplegar un flujo de mensajes para leer mensajes desde el puerto TCP/IP en el cual escucha un intermediario y publica los mensajes.

   Puede hacerlo de cualquiera de estas formas:

   - Crear un flujo de mensajes que contenga un nodo de proceso de mensajes **Real-timeOptimizedFlow**.
   - Crear un flujo de mensajes que contenga un nodo de proceso de mensajes **Real-timeInput** y un nodo de proceso de mensajes Publication.

   Debe configurar el nodo **Real-timeOptimizedFlow** o **Real-timeInput** para escuchar en el puerto utilizado para conexiones en tiempo real. En XMS, el número de puerto predeterminado para conexiones en tiempo real es 1506.

   No es necesario que realice este paso si ya se ha desplegado un flujo de mensajes apropiado en el intermediario.
2. Si requiere que los mensajes se entreguen en la aplicación mediante WebSphere MQ Multicast Transport, configure el intermediario para habilitar la multidifusión. Configure los temas que deben tener la multidifusión habilitada, especificando una calidad de servicio fiable para estos temas que requieren una multidifusión fiable.
3. Si la aplicación proporciona un ID de usuario y una contraseña cuando se conecta a un intermediario, y desea que el intermediario autentique su aplicación utilizando esta información, configure el servidor de nombres de usuario y el intermediario para una autenticación de contraseña simple similar al telnet.

Resultados

Ahora puede iniciar la aplicación.

Tareas relacionadas

Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ
En esta sección se presupone que está utilizando IBM WebSphere MQ versión 7.0. Para poder ejecutar una aplicación que se conecta a un IBM WebSphere MQ gestor de colas, debe configurar el gestor de colas. Para una aplicación de Publicar/Suscribir, es necesaria alguna configuración adicional si está utilizando la interfaz de publicación/suscripción en cola.

Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus
Antes de poder ejecutar una aplicación que se conecta a un WebSphere Servicio Integration Bus, debe configurar el bus de integración de servicios del mismo modo que configura el bus de integración de servicios para ejecutar aplicaciones JMS que utilizan el proveedor de mensajería predeterminado.

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Referencia relacionada

Requisitos previos para las aplicaciones XMS que se conectan a IBM WebSphere MQ
Algunos requisitos previos se aplican si la aplicación XMS se conecta a IBM WebSphere MQ.

Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus

Antes de poder ejecutar una aplicación que se conecta a un WebSphere Servicio Integration Bus, debe configurar el bus de integración de servicios del mismo modo que configura el bus de integración de servicios para ejecutar aplicaciones JMS que utilizan el proveedor de mensajería predeterminado.

Antes de empezar

Antes de iniciar esta tarea, debe realizar los pasos siguientes:

- Asegúrese de que se crea un bus de mensajería y que el servidor se ha añadido al bus como miembro del bus.
- Asegúrese de que la aplicación tiene acceso a un bus de integración de servicios que contiene, como mínimo, un motor de mensajería en ejecución.
- Si es necesario el funcionamiento de HTTP, se debe definir un canal de transporte de entrada de motor de mensajería de HTTP. De forma predeterminada, los canales para SSL y TCP se definen durante la instalación del servidor.
- Asegúrese de que la aplicación utiliza una fábrica de conexiones cuyas propiedades se han definido correctamente para conectar con el bus de integración de servicios utilizando un servidor de programa de arranque. La información mínima necesaria es:
  - El punto final de proveedor, que describe la ubicación y el protocolo para utilizar al negociar una conexión al servidor de mensajería (es decir, a través del servidor de programa de arranque). En su forma más simple, para un servidor instalado con valores predeterminados, el punto final de proveedor se puede definir de forma que sea el nombre de host del servidor.
  - El nombre del bus a través del cual se envían los mensajes.

  Si desea más información sobre las propiedades de una fábrica de conexiones, consulte "Propiedades de ConnectionFactory" en la página 185.

Acerca de esta tarea

Cualquier cola o espacio de tema que necesite debe estar definido. De forma predeterminada, un espacio de tema llamado Default.Topic.Space se define durante la instalación del servidor pero, si necesita más espacios de tema, debe crear estos espacios de tema usted mismo. No es necesario predefinir temas individuales en un espacio de tema, porque el servidor crea una instancia de estos temas individuales de forma dinámica, según sea necesario.

Los pasos siguientes resumen lo qué debe hacer.

Procedimiento

1. Cree las colas que la aplicación necesite para la mensajería punto a punto.
2. Cree cualquier espacio de tema adicional que la aplicación necesite para la mensajería de Publicar/Suscribir.

Resultados

Ahora puede iniciar la aplicación.

Tareas relacionadas

Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ
En esta sección se presupone que está utilizando IBM WebSphere MQ versión 7.0. Para poder ejecutar una aplicación que se conecta a un IBM WebSphere MQ gestor de colas, debe configurar el gestor de colas. Para una aplicación de Publicar/Suscribir, es necesaria alguna configuración adicional si está utilizando la interfaz de publicación/suscripción en cola.

Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario
Antes de poder ejecutar una aplicación que utiliza una conexión en tiempo real con un intermediario, debe configurar ese intermediario.

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Referencia relacionada

Requisitos previos para las aplicaciones XMS que se conectan a IBM WebSphere MQ
Algunos requisitos previos se aplican si la aplicación XMS se conecta a IBM WebSphere MQ.

Instalación de Message Service Client for .NET mediante el asistente de instalación

La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Acerca de esta tarea

Para instalar Message Service Client for .NET en Windows, siga este procedimiento.

Procedimiento

1. Si esta instalando desde un SupportPac, complete los pasos siguientes, de lo contrario,vaya directamente al paso “2” en la página 16.
   a) En Windows, inicie sesión como administrador.
   b) Ejecute el instalador dotNETClientsetup.exe.
2. Espere a que se abra el asistente de instalación y aparezca el mensaje siguiente:

   ```
   Welcome to IBM Message Service Client for .NET installation wizard
   ```

   Pulse **Siguiente**.

   El asistente le podría solicitar que lea el acuerdo de licencia.
3. Si se le ha solicitado leer el acuerdo de licencia y acepta los términos del acuerdo de licencia, pulse **Acepto los términos del acuerdo de licencia** y, a continuación, pulse **Siguiente**.

   El asistente de instalación le solicita que elija el tipo de configuración que mejor se adapte a sus necesidades.
4. Seleccione el tipo de configuración que necesita:
   - Para instalar todas las características de programa, e instalarlas en el directorio de instalación predeterminado, pulse **Completar**.
   - Para elegir qué características desea instalar y especificar dónde se instalan, pulse **Personalizar**.
5. Pulse **Siguiente**.

   Si selecciona la opción de instalación completa, el asistente de instalación muestra un mensaje que indica que está preparado para iniciar la instalación, tal como se describe en el paso “8” en la página 17. Si selecciona la opción de instalación personalizada, el asistente de instalación le solicita que seleccione las características que desea instalar y debe completar el paso “6” en la página 16 y el paso “7” en la página 16 antes de pasar al paso “8” en la página 17.
6. Solo para una instalación personalizada, pulse un icono en la lista de características para especificar los cambios sobre cómo desea que se instalen las características de Message Service Client for .NET. Si no desea instalar Message Service Client for .NET en el directorio sugerido, elija otro directorio.

   Si elige instalar Message Service Client for .NET en un directorio que actualmente no existe, el asistente de instalación crea el directorio automáticamente.

   Si desea desarrollar aplicaciones XMS, asegúrese de que está seleccionada la característica **Ejemplos y herramientas de desarrollo**. Esta característica proporciona las aplicaciones de ejemplo, y las bibliotecas y cualquier otro archivo necesario para compilar aplicaciones .NET. Si no selecciona esta característica, solo se instalan los archivos necesarios para ejecutar aplicaciones XMS.
7. Si está utilizando la opción de instalación personalizada, pulse **Siguiente** tras seleccionar las opciones que necesita, tal como se describe en el paso “6” en la página 16.

El asistente de instalación muestra un mensaje que indica que está preparado para iniciar la instalación.

8. Pulse **Instalar** para iniciar la instalación.

   El asistente de instalación muestra una barra que muestra el progreso de la instalación. Espere a que se complete la barra de progreso. Cuando la instalación se completa correctamente, la ventana muestra el mensaje siguiente:

   ```
   The installation wizard has successfully installed IBM Message Service Client
   for .NET. Click Finish to exit the wizard.
   ```

9. Pulse **Finalizar** para cerrar el asistente de instalación.

Resultados

Ha instalado correctamente Message Service Client for .NET, que está preparado para ser utilizado.

Qué hacer a continuación

Antes de ejecutar cualquier aplicación XMS, incluyendo las aplicaciones de ejemplo proporcionadas con XMS, debe configurar el entorno del servidor de mensajería, si desea más detalles, consulte: “Configuración del entorno de servidor de mensajería” en la página 11.

Conceptos relacionados

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Configuración del entorno de servidor de mensajería
Este secciónCapítulo describe cómo configurar el entorno del servidor de mensajería para permitir que las aplicaciones XMS se conecten a un servidor.

Utilización de las aplicaciones de ejemplo XMS
Utilice las aplicaciones de ejemplo proporcionadas con XMS para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones. Los ejemplos proporcionan una descripción general de las características comunes de cada API.

Tareas relacionadas

Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ
En esta sección se presupone que está utilizando IBM WebSphere MQ versión 7.0. Para poder ejecutar una aplicación que se conecta a un IBM WebSphere MQ gestor de colas, debe configurar el gestor de colas. Para una aplicación de Publicar/Suscribir, es necesaria alguna configuración adicional si está utilizando la interfaz de publicación/suscripción en cola.

Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario
Antes de poder ejecutar una aplicación que utiliza una conexión en tiempo real con un intermediario, debe configurar ese intermediario.

Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus
Antes de poder ejecutar una aplicación que se conecta a un WebSphere Servicio Integration Bus, debe configurar el bus de integración de servicios del mismo modo que configura el bus de integración de servicios para ejecutar aplicaciones JMS que utilizan el proveedor de mensajería predeterminado.

Referencia relacionada

Requisitos previos para las aplicaciones XMS que se conectan a IBM WebSphere MQ
Algunos requisitos previos se aplican si la aplicación XMS se conecta a IBM WebSphere MQ.

Requisitos previos para las aplicaciones XMS que se conectan a IBM WebSphere MQ

Algunos requisitos previos se aplican si la aplicación XMS se conecta a IBM WebSphere MQ.

Para aplicaciones que se conectan a un gestor de colas IBM WebSphere MQ, debe instalar las bibliotecas de cliente apropiadas de IBM WebSphere MQ en la máquina que utiliza para ejecutar la aplicación XMS. Estas bibliotecas están preinstaladas en máquinas con un gestor de colas local.

Para Cliente XMS para .NET, utilice las bibliotecas de cliente que se entregan con IBM WebSphere MQ Versión 7.0.1.0 o posterior. Estas son las clases *IBM WebSphere MQ para .NET*. Se permiten conexiones de modalidad de cliente a gestores de colas IBM WebSphere MQ Versión 7.0, IBM WebSphere MQ Versión 6.0 y IBM WebSphere MQ Versión 5.3 y conexiones de modalidad de enlaces a un gestor de colas local, si también la versión 7.0.1.0 o posterior.

Microsoft .NET Framework Versión 2.0 Redistributable Package debe estar instalado en el sistema en el que se va a instalar XMS . Si esta paquete no está disponible, la instalación de XMS fallará. A continuación, debe salir del procedimiento de instalación, instalar Microsoft .NET Framework Versión 2.0 Redistributable Package en el sistema y volver a ejecutar el procedimiento de instalación.

En el sitio de descarga de Microsoft, debe buscar dotnetfx.exe para Microsoft .NET Framework Versión 2.0 Paquete redistribuible (x86) y NetFx64.exe para Microsoft .NET Framework Versión 2.0 Paquete redistribuible (x64), según sea aplicable.

Conceptos relacionados

"Configuración del entorno de servidor de mensajería" en la página 11
Este secciónCapítulo describe cómo configurar el entorno del servidor de mensajería para permitir que las aplicaciones XMS se conecten a un servidor.

Tareas relacionadas

Configuración del gestor de colas y del intermediario para una aplicación que conecta con un gestor de colas de IBM WebSphere MQ
En esta sección se presupone que está utilizando IBM WebSphere MQ versión 7.0. Para poder ejecutar una aplicación que se conecta a un IBM WebSphere MQ gestor de colas, debe configurar el gestor de colas. Para una aplicación de Publicar/Suscribir, es necesaria alguna configuración adicional si está utilizando la interfaz de publicación/suscripción en cola.

Configuración de un intermediario para una aplicación que utiliza una conexión en tiempo real con un intermediario
Antes de poder ejecutar una aplicación que utiliza una conexión en tiempo real con un intermediario, debe configurar ese intermediario.

Configuración del bus de integración de servicios para una aplicación que se conecta a un WebSphere Servicio Integration Bus
Antes de poder ejecutar una aplicación que se conecta a un WebSphere Servicio Integration Bus, debe configurar el bus de integración de servicios del mismo modo que configura el bus de integración de servicios para ejecutar aplicaciones JMS que utilizan el proveedor de mensajería predeterminado.

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Utilización de las aplicaciones de ejemplo XMS

Utilice las aplicaciones de ejemplo proporcionadas con XMS para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones. Los ejemplos proporcionan una descripción general de las características comunes de cada API.

Conceptos relacionados

"Las aplicaciones de ejemplo" en la página 19
Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

Tareas relacionadas

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

"Ejecución de las aplicaciones de ejemplo" en la página 20
Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

"Creación de aplicaciones de ejemplo de .NET" en la página 21
Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

Las aplicaciones de ejemplo

Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

Si necesita ayuda para crear sus propias aplicaciones, puede utilizar las aplicaciones de ejemplo como punto de partida. Se proporcionan ambas versiones, la de origen y la compilada, para cada aplicación. Revise el código fuente de ejemplo e identifique los pasos clave para crear cada objeto necesario para la aplicación (ConnectionFactory, Connection, Session, Destination, y un Producer, or un Consumer, o ambos), y para establecer cualquier propiedad específica necesaria para especificar cómo desea que funcione la aplicación. Si desea más información, consulte "Cómo escribir aplicaciones de XMS" en la página 22. Los ejemplos están sujetos a cambios en futuros releases de XMS.

La tabla siguiente muestra los tres conjuntos de aplicaciones de ejemplo (uno para cada API) que se proporcionan con XMS.

Nombre de ejemplo	Descripción
SampleConsumerCS	Una aplicación de consumidor de mensajes que toma mensajes de una cola o que se suscribe a un tema.
SampleProducerCS	Una aplicación de productor de mensajes que genera mensajes en una cola o sobre un tema.
SampleConfigCS	Una aplicación de configuración que puede utilizar para crear un repositorio de objetos administradores que se basa en archivo. La aplicación contiene una fábrica de conexiones y un destino para sus valores de conexión en particular. Este repositorio de objetos administrados se puede utilizar después con cada una de las aplicaciones de productor y consumidor de ejemplo.

Los ejemplos que dan soporte a las mismas funciones den las distintas API tienen diferencias sintácticas.

- Las aplicaciones de productor y consumidor de mensajes de ejemplo soportan ambas las funciones siguientes:
  - Conexiones con IBM WebSphere MQ, WebSphere Event Broker, WebSphere Message Broker (utilizando una conexión en tiempo real con un intermediario) y un WebSphere Servicio Integration Bus
  - Búsquedas de repositorio de objetos administrados utilizando la interfaz de contexto inicial
  - Conexiones a colas (IBM WebSphere MQ y WebSphere Servicio Integration Bus) y temas (IBM WebSphere MQ, conexión en tiempo real a un intermediario, y WebSphere Servicio Integration Bus)
  - Mensajes base, de byte, de correlación, de objeto, de corriente de datos y de texto
- La aplicación de consumidor de mensajes de ejemplo admite las modalidades de recepción síncrona y asíncrona y sentencias de SQL Selector.

- La aplicación de productor de mensajes de ejemplo admite las modalidades de entrega persistente y no persistente.

Modalidades de funcionamiento

Los ejemplos pueden funcionar de una de estas dos modalidades:

Modalidad simple
Puede ejecutar los ejemplos con la mínima entrada de usuario.

Modalidad avanzada
Puede personalizar de forma más precisa la forma en la que funcionan los ejemplos.

Todos los ejemplos son compatibles y, por lo tanto, pueden funcionar entre lenguajes.

Dónde encontrar los ejemplos

Para averiguar dónde están instaladas las aplicaciones de ejemplo para Message Service Client for .NET , consulte *Directorios instalados en Windows (.NET)* en la documentación del producto en línea de IBM IBM WebSphere MQ .

Conceptos relacionados

"Creación de sus propias aplicaciones" en la página 45
Cree sus propias aplicaciones como crea las aplicaciones de ejemplo.

Tareas relacionadas

Ejecución de las aplicaciones de ejemplo
Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

Creación de aplicaciones de ejemplo de .NET
Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

"Ejecución de las aplicaciones de ejemplo" en la página 20
Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

"Creación de aplicaciones de ejemplo de .NET" en la página 21
Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

Ejecución de las aplicaciones de ejemplo

Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

Antes de empezar

Antes de ejecutar cualquiera de las aplicaciones de ejemplo proporcionadas, primero debe configurar el entorno del servidor de mensajería para que las aplicaciones se puedan conectar a un servidor. Consulte "Configuración del entorno de servidor de mensajería" en la página 11.

Procedimiento

Para ejecutar una aplicación de ejemplo .NET, complete los pasos siguientes:

Consejo: Cuando se ejecuta una aplicación de ejemplo, escriba ? en cualquier momento para obtener ayuda sobre qué hacer a continuación.

1. Seleccione la modalidad en la cual desea ejecutar la aplicación de ejemplo.
   Escriba `Advanced` o `Simple`.
2. Responda las preguntas.

Para seleccionar el valor predeterminado, que se muestra entre corchetes al final de la pregunta, pulse Intro. Para seleccionar un valor diferente, escriba el valor apropiado y pulse Intro.

A continuación, se muestra una pregunta de ejemplo:

```
Enter connection type [wpm]:
```

En este caso, el valor predeterminado es `wpm` (conexión a un WebSphere Servicio Integration Bus).

Resultados

Cuando se ejecutan las aplicaciones de ejemplo, los archivos de respuestas se generan automáticamente en el directorio de trabajo actual. Los nombres de archivo de respuestas tienen el formato *connection_type-sample_type*`.rsp`; por ejemplo, `wpm-producer.rsp`. Si es necesario, puede utilizar el archivo de respuestas generado para volver a ejecutar la aplicación de ejemplo con las mismas opciones, de modo que no tenga que volver a entrar las opciones.

Conceptos relacionados

Las aplicaciones de ejemplo
Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

"Las aplicaciones de ejemplo" en la página 19
Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

Tareas relacionadas

Creación de aplicaciones de ejemplo de .NET
Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

"Creación de aplicaciones de ejemplo de .NET" en la página 21
Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

Creación de aplicaciones de ejemplo de .NET

Cuando se crea una aplicación de ejemplo de .NET, se crea una versión ejecutable del ejemplo seleccionado.

Antes de empezar

Instale el compilador apropiado. Consulte *Instalación de Message Service Client for .NET* en la documentación del producto en línea de IBM IBM WebSphere MQ . Esta tarea presupone que tiene instalado Visual Studio 2005 y que está familiarizado con su uso.

Procedimiento

Para crear una aplicación de ejemplo .NET, complete los pasos siguientes:

1. Pulse el archivo de solución `Samples.sln` proporcionado con los ejemplos de .NET.
2. Pulse con el botón derecho del ratón en la solución `Samples` en la ventana del Explorador de soluciones y seleccione **Crear solución**.

Resultados

Se crea un programa ejecutable en la subcarpeta apropiada del ejemplo, `bin/Debug` o `bin/Release`, según la configuración que haya elegido. Este programa tiene el mismo nombre que la carpeta, con un sufijo de CS. Por ejemplo, si está creando la versión C# de la aplicación de ejemplo del productos de mensajes, se crea `SampleProducerCS.exe` en la carpeta `SampleProducer`.

Conceptos relacionados

Las aplicaciones de ejemplo
Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

"Las aplicaciones de ejemplo" en la página 19
Las aplicaciones de ejemplo proporcionan una descripción general de las características comunes de cada API. Puede utilizarlas para verificar la instalación y la configuración del servidor de mensajería y para ayudarle a crear sus propias aplicaciones.

"Creación de sus propias aplicaciones" en la página 45
Cree sus propias aplicaciones como crea las aplicaciones de ejemplo.

Tareas relacionadas

Ejecución de las aplicaciones de ejemplo
Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

"Ejecución de las aplicaciones de ejemplo" en la página 20
Puede ejecutar las aplicaciones de ejemplo de .NET de forma interactiva en la modalidad simple o avanzada, o de forma no interactiva utilizando archivos de respuestas auto-generados o personalizados.

Desarrollo de aplicaciones XMS

Este secciónCapítulo proporciona información que puede ser útil al escribir aplicaciones XMS.

La información de este secciónCapítulo es para aplicaciones .NET.

Si desea más información sobre cómo escribir aplicaciones XMS, consulte los temas siguientes:

Cómo escribir aplicaciones de XMS

Este secciónCapítulo proporciona información para ayudarle a escribir aplicaciones XMS.

Este secciónCapítulo contiene conceptos generales para escribir aplicaciones XMS. Consulte también "Escritura de aplicaciones XMS para .NET" en la página 46 para obtener información específica para crear aplicaciones .NET.

Este secciónCapítulo contiene los temassecciones siguientes:

Referencia relacionada

Interfaces .NET

En este temasección se documentan las interfaces de clase .NET y sus propiedades y métodos.

El modelo de agrupación en hebras

Reglas generales controlan cómo una aplicación de varias hebras puede utilizar objetos XMS.

- Solo los objetos de los tipos siguientes se pueden utilizar de forma simultánea en hebras distintas:
  - ConnectionFactory
  - Conexión
  - ConnectionMetaData
  - Destino
- Un objeto Session se puede utilizar en solo una sola hebra a la vez.

Las excepciones a estas reglas están indicadas por las entradas etiquetadas como "Contexto de hebras" en las definiciones de interfaz de los métodos en los capítulos de referencia de la API“Referencia de Message Service Clients para .NET” en la página 92.

Condiciones de error que se pueden manejar durante el tiempo de ejecución

Los códigos de retorno de llamadas de API son condiciones de error que se pueden manejar durante el tiempo de ejecución. La forma en la cual gestiona este tipo de error depende de si está utilizando la API C o C++.

Cómo detectar errores durante el tiempo de ejecución

Si una aplicación llama a una función de API C y la llamada falla, se devuelve una respuesta con un código de retorno distinto a XMS_OK con un bloque de error XMS que contiene más información sobre la razón de la anomalía.

La API C++ lanza una excepción cuando se utiliza un método.

Una aplicación utiliza una escucha de excepción para que se le notifique de forma asíncrona un problema con una conexión. El escucha de excepción se proporciona a, y se inicializa utilizando, la API C XMS o C++.

Cómo manejar errores durante el tiempo de ejecución

Algunas condiciones de error son una indicación de que algún recurso no está disponible, y la acción que puede realizar una aplicación depende de la función XMS a la que está llamando la aplicación. Por ejemplo, si una conexión no se puede conectar al servidor, la aplicación podría desear intentarlo periódicamente hasta que se realice una conexión. Un bloque de error o una excepción de XMS podría no contener información suficiente para determinar qué acción realizar y, en estas situaciones, a menudo, hay un bloque de error o excepción enlazado que contiene información de diagnóstico más específica.

En la API C, pruebe siempre una respuesta con un código de retorno distinto a XMS_OK y pase siempre un bloque de error en la llamada de API. Normalmente, la acción realizada depende de la función de API que está utilizando la aplicación.

En la API C++, incluya siempre llamadas a métodos en un bloque Try y para captar todos los tipos de excepción de XMS, yespecifique la clase Exception en la construcción de captación.

El escucha de excepción es una vía de acceso de condición de error asíncrona que se puede iniciar en cualquier momento. Cuando se inicia la función de escucha de excepción, en su propia hebra, normalmente, es una indicación de una anomalía más grave que una condición de error de API XMS normal. Se puede realizar cualquier acción apropiada, pero se debe seguir atentamente las reglas para el modelo de hebras de XMS, tal como se describe en “El modelo de agrupación en hebras” en la página 23.

Objetos ConnectionFactories y Connection

Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Para .NET, la aplicación Un XMS primero utiliza un objeto XMSFactoryFactory para obtener una referencia a un objeto ConnectionFactory que sea apropiado para el tipo de protocolo necesario. Este objeto ConnectionFactory puede generar conexiones solo para ese tipo de protocolo.

La aplicación Un XMS puede crear varias conexiones y una aplicación con varias hebras puede utilizar un solo objeto Connection de forma simultánea en varias hebras. Un objeto Connection encapsula una conexión de comunicaciones entre una aplicación y un servidor de mensajería.

Una conexión sirva para varias finalidades:

- Cuando una aplicación crea una conexión, la aplicación se puede autenticar.
- Una aplicación puede asociar un identificador de cliente exclusivo a una conexión. El identificador de cliente se utiliza para dar soporte a las suscripciones duraderas en el dominio de Publicar/Suscribir. El identificador de cliente se puede establecer de dos maneras:

  La forma preferida de asignar un identificador de cliente de conexiones es configurar en un objeto ConnectionFactory específico de cliente utilizando propiedades y asignarlo de forma transparente a la conexión que crea.

  Una forma alternativa de asignar un identificador de cliente es utilizar un valor específico del proveedor que se establece en el objeto Connection. Este valor no sustituye el identificador que se ha configurado de forma administrativa. Se proporciona para el caso donde no existe ningún identificador especificado de forma administrativa. Si un identificador especificado de forma administrativa no existe, un intento de sustituirlo con un valor específico de proveedor provoca que se lance una excepción. Si una aplicación define de forma explícita un identificador, debe hacerlo inmediatamente después de crear la conexión y antes de que se realice cualquier otra acción en la conexión; de lo contrario, se lanza una excepción.

Normalmente, una aplicación Un XMS crea una conexión, una o más sesiones y varios productores y consumidores de mensajes.

La creación de una conexión es relativamente cara en términos de recursos del sistema, porque implica el establecimiento de una conexión de comunicaciones y, también, podría implicar la autenticación de la aplicación.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Modalidad iniciada y detenida de conexión

Una conexión puede operar tanto en modalidad iniciada como detenida.

Cuando una aplicación crea una conexión, la conexión está en la modalidad detenida. Cuando la conexión está en la modalidad detenida, la aplicación puede inicializar sesiones, y puede enviar mensajes pero no puede recibirlos, ya sea de forma síncrona o asíncrona.

Una aplicación puede iniciar una conexión llamando al método `Start Connection` . Cuando la conexión está en la modalidad iniciada, la aplicación puede enviar y recibir mensajes. A continuación, la aplicación puede detener y reiniciar la conexión llamando a los métodos Detener conexión y `Start Connection` .

Conceptos relacionados

Cierre de conexión
Una aplicación cierra una conexión llamando al método Cerrar conexión.

Manejo de excepciones
Si una aplicación utiliza una conexión solo para consumir mensajes de forma asíncrona, obtiene información sobre un problema con la conexión solo utilizando un escucha de excepción.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Cierre de conexión

Una aplicación cierra una conexión llamando al método Cerrar conexión.

Cuando una aplicación cierra una conexión, XMS realiza las acciones siguientes:

- Cierra todas las sesiones asociadas a la conexión y suprime determinados objetos asociados a estas sesiones. Si desea más información sobre qué objetos se suprimen, consulte “Supresión de objeto” en la página 40. Al mismo tiempo, XMS retrotrae las transacciones actualmente en curso dentro de las sesiones.
- Finaliza la conexión de comunicaciones con el servidor de mensajería.
- Libera la memoria y otros recursos internos utilizados por la conexión.

XMS no crea ningún acuse de recibo de la recepción de ninguno de los mensajes que no ha podido reconocer durante una sesión, antes de cerrar la conexión. Si desea más información sobre cómo crear un acusar recibo de la recepción de mensajes, consulte “Acuse de recibo de mensaje” en la página 28.

Conceptos relacionados

Modalidad iniciada y detenida de conexión
Una conexión puede operar tanto en modalidad iniciada como detenida.

Manejo de excepciones
Si una aplicación utiliza una conexión solo para consumir mensajes de forma asíncrona, obtiene información sobre un problema con la conexión solo utilizando un escucha de excepción.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Manejo de excepciones

Si una aplicación utiliza una conexión solo para consumir mensajes de forma asíncrona, obtiene información sobre un problema con la conexión solo utilizando un escucha de excepción.

Todas las excepciones de XMS .NET se han derivado de System.Exception. Si desea más información, consulte “Manejo de errores en .NET” en la página 50.

Conceptos relacionados

Modalidad iniciada y detenida de conexión
Una conexión puede operar tanto en modalidad iniciada como detenida.

Cierre de conexión
Una aplicación cierra una conexión llamando al método Cerrar conexión.

Conexión con un bus de integración de servicios WebSphere

La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Conexión con un bus de integración de servicios WebSphere

La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

El protocolo HTTP se puede utilizar en situaciones donde no es posible una conexión TCP/IP directa. Una situación común es cuando se comunican a través de un cortafuegos como, por ejemplo, cuando dos empresas intercambian mensajes. A menudo, el uso de HTTP para comunicarse a través de un cortafuegos se denomina *ejecución en túnel HTTP* Sin embargo, la ejecución en túnel HTTP es inherentemente más lenta que el uso de la conexión TCP/IP directa porque las cabeceras HTTP añaden una cantidad de datos significativa que se transfieren y porque el protocolo HTTP requiere más flujos de comunicación que TCP/IP.

Para crear una conexión TCP/IP, una aplicación puede utilizar una fábrica de conexiones cuya propiedad XMSC_WPM_TARGET_TRANSPORT_CHAIN está establecida en XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC. Este es el valor predeterminado de la propiedad. Si la conexión se crea correctamente, la propiedad XMSC_WPM_CONNECTION_PROTOCOL de la conexión se establece en XMSC_WPM_CP_TCP.

Para crear una conexión que utiliza HTTP, una aplicación debe utilizar una fábrica de conexiones cuya propiedad XMSC_WPM_TARGET_TRANSPORT_CHAIN se establece en el nombre de una cadena de transporte de salida, que se ha configurado para utilizar un canal de transporte HTTP. Si la conexión se crea correctamente, la propiedad XMSC_WPM_CONNECTION_PROTOCOL de la conexión se establece en XMSC_WPM_CP_HTTP. Para obtener información sobre cómo configurar cadenas de transporte, consulte Cadenas de transporte en la duplicación de WebSphere Application Server .

Una aplicación tiene una opción similar de protocolos de comunicaciones al conectarse a un servidor de programa de arranque. La propiedad XMSC_WPM_PROVIDER_ENDPOINTS de una fábrica de conexiones es una secuencia de una o más direcciones de punto final de servidores de programa de arranque. El componente de cadena de transporte del programa de arranque de cada dirección de punto final puede ser XMSC_WPM_BOOTSTRAP_TCP, para una conexión de TCP/IP a un servidor de programa de arranque o XMSC_WPM_BOOTSTRAP_HTTP, para una conexión que utiliza HTTP.

Conceptos relacionados

Modalidad iniciada y detenida de conexión
Una conexión puede operar tanto en modalidad iniciada como detenida.

Cierre de conexión
Una aplicación cierra una conexión llamando al método Cerrar conexión.

Manejo de excepciones
Si una aplicación utiliza una conexión solo para consumir mensajes de forma asíncrona, obtiene información sobre un problema con la conexión solo utilizando un escucha de excepción.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Sesiones

Una sesión es un contexto de hebra única para enviar y recibir mensajes.

Una aplicación puede utilizar una sesión para crear mensajes, productores de mensajes, consumidores de mensajes, navegadores de colas y destinos temporales. Una aplicación también puede utilizar una sesión para ejecutar transacciones locales.

Una aplicación puede crear varias sesiones, donde cada sesión produce y consume mensajes independientemente de las otras sesiones. Si dos consumidores de mensajes en sesiones separadas (o incluso en la misma sesión) se suscriben al mismo tema, cada uno recibe una copia de cualquier mensaje publicado sobre dicho tema.

A diferencia de un objeto Connection, un objeto Session no se puede utilizar de forma simultánea en hebras diferentes. Solo el método Cerrar sesión de un objeto Session se puede llamar desde una hebra distinta a la que está utilizando el objeto Session en ese momento. El método Cerrar sesión finaliza una sesión y libera cualquier recurso del sistema asignado a la sesión.

Si una aplicación debe procesar mensajes de forma simultánea en más de una hebra, la aplicación debe crear una sesión en cada hebra y, después, utilizar esa sesión para cualquier operación enviar o recibir en dicha hebra.

Sesiones con transacción

Las aplicaciones XMS pueden ejecutar transacciones locales. Una *transacción local* es una transacción que implica cambios solo en los recursos del gestor de colas o el bus de integración de servicios al cual está conectada la aplicación.

La información de este temasección solo es pertinente si una aplicación se conecta a un gestor de colas de IBM WebSphere MQ o a un bus de integración de servicios de WebSphere. La información no es relevante para una conexión en tiempo real con un intermediario.

Para ejecutar transacciones locales, en primer lugar, una aplicación debe crear una sesión con transacción llamando al método Crear sesión de un objeto Connection, especificando como parámetro que la sesión es una sesión con transacción. Por consiguiente, todos los mensajes enviados y recibidos dentro de la sesión se agrupan en una secuencia de transacciones. Una transacción finaliza cuando la aplicación confirma o retrotrae los mensajes que ha enviado y recibido desde que empezó la transacción.

Para confirmar una transacción, una aplicación llama al método Confirmar del objeto Session. Cuando se confirma una transacción, todos los mensajes enviados en la transacción pasan a estar disponibles para su entrega a otras aplicaciones, y todos los mensajes recibidos en la transacción reciben el acuse de recibo, de forma que el servidor de mensajería no los intenta volver a entregar a la aplicación. En el dominio punto a punto, el servidor de mensajería también elimina los mensajes recibidos de sus colas.

Para retrotraer una transacción, una aplicación llama al método Retrotraer del objeto Session. Cuando una transacción se retrotrae, el servidor de mensajería descarta todos los mensajes enviados en la transacción y todos los mensajes recibidos en la transacción pasan a estar disponibles para volverlos a entregar. En el dominio punto a punto, los mensajes que se han recibido se vuelven a colocar en sus colas y vuelven a ser visibles para las demás aplicaciones.

Una nueva transacción se inicia automáticamente cuando una aplicación crea una sesión con transacción o llama al método Confirmar o Retrotraer. Por lo tanto, una sesión con transacción siempre tiene una transacción activa.

Cuando una aplicación cierra una sesión con transacción, se produce una retrotracción implícita. Cuando una aplicación cierra una conexión, se produce una retrotracción implícita para todas las sesiones con transacción de la conexión.

Una transacción está incluida íntegramente en una sesión con transacción. Una transacción no puede abarcar sesiones. Esto significa que no es posible para una aplicación enviar y recibir mensajes en dos

o más sesiones con transacción y, después, confirmar o retrotraer todas estas acciones como una sola transacción.

Conceptos relacionados

Acuse de recibo de mensaje
Cada sesión que no es transaccional tiene una modalidad de acuse de recibo que determina cómo se acusa recibo de los mensajes recibidos por la aplicación. Existen tres modalidades de acuse de recibo disponibles, y la elección de la modalidad afecta al diseño de la aplicación.

Entrega de mensajes asíncrona
XMS utiliza una hebra para manejar todas las entregas de mensajes asíncronas para una sesión. Esto significa que solo se puede ejecutar una función de escucha de mensajes o un método onMessage() a la vez.

Entrega de mensajes síncrona
Los mensajes se entregan de forma síncrona a una aplicación si la aplicación utiliza los métodos Recibir de objetos MessageConsumer.

Modalidad de entrega de mensajes
XMS admite dos modalidades de entrega de mensajes.

Acuse de recibo de mensaje

Cada sesión que no es transaccional tiene una modalidad de acuse de recibo que determina cómo se acusa recibo de los mensajes recibidos por la aplicación. Existen tres modalidades de acuse de recibo disponibles, y la elección de la modalidad afecta al diseño de la aplicación.

La información de este temasección solo es pertinente si una aplicación se conecta a un gestor de colas de IBM WebSphere MQ o a un WebSphere Servicio Integration Bus. La información no es relevante para una conexión en tiempo real con un intermediario.

XMS utiliza el mismo mecanismo para acusar recibo de mensajes que utiliza JMS.

Si una sesión no es transaccional, la forma en que se acusa recibo de los mensajes recibidos por la aplicación se determina mediante la modalidad de acuse de recibo de la sesión. En los párrafos siguientes se describen las tres modalidades de acuse de recibo:

XMSC_AUTO_ACKNOWLEDGE
La sesión acusa recibo automáticamente de cada mensaje recibido por la aplicación.

Si los mensajes se entregan de forma síncrona a la aplicación, la sesión acusa recibo de cada mensaje cada vez que se completa una llamada Receive.

Si la aplicación recibe un mensaje correctamente, pero un error impide que se realice el acuse de recibo, el mensaje pasa de nuevo a estar disponible para su entrega. Por lo tanto, la aplicación debe poder ser capaz de manejar un mensaje que se vuelve a entregar.

XMSC_DUPS_OK_ACKNOWLEDGE
La sesión acusa recibo de los mensajes recibidos por la aplicación en momentos que selecciona.

Si se utiliza esta modalidad de acuse de recibo se reduce la cantidad de trabajo que debe realizar la sesión, pero un error que impida el acuse de recibo del mensaje podría provocar que más de un mensaje pase a estar disponible para ser entregado de nuevo. Por lo tanto, la aplicación debe poder ser capaz de manejar los mensajes que se vuelven a entregar.

XMSC_CLIENT_ACKNOWLEDGE
La aplicación acusa recibo de los mensajes que recibe llamando al método Acusar recibo de la clase Message.

La aplicación acusa recibo de cada mensaje de forma individual, o puede recibir un lote de mensajes y llamar al método Acusar recibo solo para el último mensaje que recibe. Cuando se llama al método Acusar recibo, se acusará recibo de todos los mensajes recibidos desde la última vez que se llamó al método.

Conjuntamente con cualquiera de estas modalidades de acuse de recibo, una aplicación puede detener y reiniciar la entrega de mensajes en una sesión llamando al método Recover de la clase Session. Se

vuelven a entregar los mensajes de los que ya se había acusado recibo previamente. Sin embargo, podría ser que no se entregaran en la misma secuencia en la que se habían entregada anteriormente. Mientras tanto, podrían haber llegado los mensajes con prioridad superior y algunos de los mensajes originales podrían haber caducado. En el dominio punto a punto, algunos de los mensajes originales podrían haber sido consumidos por otra aplicación.

Una aplicación puede determinar si un mensaje se está volviendo a entregar examinando el contenido del campo de cabecera JMSRedelivered del mensaje. La aplicación lo hace llamando al método Obtener JMSRedelivered de la clase Message.

Conceptos relacionados

Sesiones con transacción
Las aplicaciones XMS pueden ejecutar transacciones locales. Una *transacción local* es una transacción que implica cambios solo en los recursos del gestor de colas o el bus de integración de servicios al cual está conectada la aplicación.

Entrega de mensajes asíncrona
XMS utiliza una hebra para manejar todas las entregas de mensajes asíncronas para una sesión. Esto significa que solo se puede ejecutar una función de escucha de mensajes o un método onMessage() a la vez.

Entrega de mensajes síncrona
Los mensajes se entregan de forma síncrona a una aplicación si la aplicación utiliza los métodos Recibir de objetos MessageConsumer.

Modalidad de entrega de mensajes
XMS admite dos modalidades de entrega de mensajes.

Entrega de mensajes asíncrona

XMS utiliza una hebra para manejar todas las entregas de mensajes asíncronas para una sesión. Esto significa que solo se puede ejecutar una función de escucha de mensajes o un método onMessage() a la vez.

Si más de un consumidor de mensajes de una sesión está recibiendo mensajes de forma asíncrona, y una función de escucha de mensajes o un método onMessage() está entregando un mensaje a un consumidor de mensajes, cualquier otro consumidor de mensajes que está esperando el mismo mensaje debe seguir esperando. Otros mensajes que están esperando a ser entregados en la sesión también deben seguir esperando.

Si una aplicación requiere una entrega simultánea de mensajes, cree más de una sesión de forma que XMS utilice más de una hebra para manejar la entrega de mensajes asíncrona. De esta forma, se pueden ejecutar de forma simultánea más de una función de escucha de mensajes o de un método onMessage().

Una sesión no se convierte en asíncrona asignado un escucha de mensajes a un consumidor. Una sesión se convierte en asíncrona solo cuando se llama al método Connection.Start. Todas las llamadas síncronas están permitidas hasta que se llama al método Connection.Start. La entrega de mensajes a consumidores se inicia cuando se llama a Connection.Start .

Si las llamadas síncronas, como la creación de un consumidor o productor, se deben realizar en una sesión asíncrona, se debe llamar a Connection.Stop . Una sesión se puede reanudar llamando al método Connection.Start para iniciar la entrega de mensajes. La única excepción a esto es la hebra de entrega de mensajes Session, que es la hebra que entrega mensajes a la función de devolución de llamada. Esta hebra puede realizar cualquier llamada en una sesión (excepto una llamada Close) en la función de devolución de llamada de mensaje.

Nota: En la modalidad no gestionada, la llamada MQDISC dentro de una función de devolución de llamada no está soportada por el cliente WMQ.NET. De esta forma, la aplicación cliente no puede Crear o Cerrar sesiones en la devolución de llamada MessageListener en la modalidad de recepción Asíncrona. Cree y elimine la sesión fuera del método MessageListener.

Conceptos relacionados

Sesiones con transacción

Las aplicaciones XMS pueden ejecutar transacciones locales. Una *transacción local* es una transacción que implica cambios solo en los recursos del gestor de colas o el bus de integración de servicios al cual está conectada la aplicación.

Acuse de recibo de mensaje
Cada sesión que no es transaccional tiene una modalidad de acuse de recibo que determina cómo se acusa recibo de los mensajes recibidos por la aplicación. Existen tres modalidades de acuse de recibo disponibles, y la elección de la modalidad afecta al diseño de la aplicación.

Entrega de mensajes síncrona
Los mensajes se entregan de forma síncrona a una aplicación si la aplicación utiliza los métodos Recibir de objetos MessageConsumer.

Modalidad de entrega de mensajes
XMS admite dos modalidades de entrega de mensajes.

Entrega de mensajes síncrona

Los mensajes se entregan de forma síncrona a una aplicación si la aplicación utiliza los métodos Recibir de objetos MessageConsumer.

Mediante los métodos Recibir, una aplicación puede esperar un mensaje un periodo de tiempo especificado, o puede esperar indefinidamente. De forma alternativa, si una aplicación no desea esperar un mensaje, puede utilizar el método Recibir sin espera.

Conceptos relacionados

Sesiones con transacción
Las aplicaciones XMS pueden ejecutar transacciones locales. Una *transacción local* es una transacción que implica cambios solo en los recursos del gestor de colas o el bus de integración de servicios al cual está conectada la aplicación.

Acuse de recibo de mensaje
Cada sesión que no es transaccional tiene una modalidad de acuse de recibo que determina cómo se acusa recibo de los mensajes recibidos por la aplicación. Existen tres modalidades de acuse de recibo disponibles, y la elección de la modalidad afecta al diseño de la aplicación.

Entrega de mensajes asíncrona
XMS utiliza una hebra para manejar todas las entregas de mensajes asíncronas para una sesión. Esto significa que solo se puede ejecutar una función de escucha de mensajes o un método `onMessage()` a la vez.

Modalidad de entrega de mensajes
XMS admite dos modalidades de entrega de mensajes.

Modalidad de entrega de mensajes

XMS admite dos modalidades de entrega de mensajes.

- Los mensajes *persistentes* se entregan solo una vez. Un servidor de mensajería toma precauciones especiales como, por ejemplo, el registro de mensajes, para garantizar que los mensajes persistentes no se pierden en el tránsito, incluso en el caso de una anomalía.
- Los mensajes *no persistentes* no se entregan más de una vez. Los mensajes no persistentes son menos fiables que los mensajes persistentes porque se pueden perder en el tránsito en caso de una anomalía.

La selección de la modalidad de entrega es una compensación entre la fiabilidad y el rendimiento. Normalmente, los mensajes no persistentes se transportan más rápidamente que los mensajes persistentes.

Conceptos relacionados

Sesiones con transacción
Las aplicaciones XMS pueden ejecutar transacciones locales. Una *transacción local* es una transacción que implica cambios solo en los recursos del gestor de colas o el bus de integración de servicios al cual está conectada la aplicación.

Acuse de recibo de mensaje

Cada sesión que no es transaccional tiene una modalidad de acuse de recibo que determina cómo se acusa recibo de los mensajes recibidos por la aplicación. Existen tres modalidades de acuse de recibo disponibles, y la elección de la modalidad afecta al diseño de la aplicación.

Entrega de mensajes asíncrona
XMS utiliza una hebra para manejar todas las entregas de mensajes asíncronas para una sesión. Esto significa que solo se puede ejecutar una función de escucha de mensajes o un método onMessage() a la vez.

Entrega de mensajes síncrona
Los mensajes se entregan de forma síncrona a una aplicación si la aplicación utiliza los métodos Recibir de objetos MessageConsumer.

Destinos

Una aplicación XMS utiliza un objeto Destination para especificar el destino de mensajes que se están enviando y el origen de mensajes que se están recibiendo.

Una aplicación XMS puede crear un objeto Destination durante el tiempo de ejecución, u obtener un destino predefinido del repositorio de objetos administrados.

Al igual que con ConnectionFactory, la forma más flexible para que una aplicación XMS especifique un destino es definirlo como un objeto administrado. Mediante este método, las aplicaciones escritas en los lenguajes C, C++, .NET y Java pueden compartir definiciones del destino. Las propiedades de objetos Destination administrados se pueden cambiar sin cambiar ningún código.

Para aplicaciones .NET, cree un destino utilizando el método CreateTopic o CreateQueue. Estos dos métodos están disponibles en ambos objetos, ISession y XMSFactoryFactory, en la API .NET. Si desea más información, consulte “Destinos en .NET” en la página 48 y “IDestination” en la página 111.

Referencia relacionada

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Identificadores uniformes de recursos de tema

El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

El URI para un tema empieza con la secuencia topic://, seguida del nombre del tema y (opcional) una lista de pares de nombre-valor que establecen las propiedades de tema restantes. Un nombre de tema no puede estar vacío.

A continuación, aparece un ejemplo en un fragmento de código de .NET:

```
topic = session.CreateTopic("topic://Sport/Football/Results?multicast=7");
```

Si desea más información sobre las propiedades de un tema, incluyendo el nombre y los valores válidos que puede utilizar en un URI, consulte “Propiedades de Destination” en la página 191.

Al especificar un URI de tema para su uso en una suscripción, se pueden utilizar los comodines. La sintaxis para estos comodines depende del tipo de conexión y de la versión de intermediario; están disponibles las opciones siguientes:

- IBM WebSphere MQ V7.0 gestor de colas con formato de comodín de nivel de carácter
- IBM WebSphere MQ V7.0 gestor de colas con formato comodín de nivel de tema
- IBM WebSphere MQ V6.0 gestor de colas con intermediario V1 (IBM WebSphere MQ V6.0 Publicar/Suscribir)

- IBM WebSphere MQ V6.0 con, o conexión en tiempo real con, el intermediario V2 (WebSphere Event Broker o WebSphere Message Broker)
- WebSphere bus de integración de servicios

IBM WebSphere MQ V7.0 gestor de colas con formato de comodín de nivel de carácter

IBM WebSphere MQ V7.0 gestor de colas con formato de comodín de nivel de carácter utiliza los siguientes caracteres comodín:

* para 0 o más caracteres
? para 1 carácter
% para un carácter de escape

Tabla 1 en la página 32 proporciona algunos ejemplos sobre cómo utilizar este esquema de comodín.

Tabla 1. Ejemplos de URI en los que se utiliza el formato comodín a nivel de carácter para el gestor de colas de IBM WebSphere MQ V7.0

Identificador uniforme de recursos	Coincide con	Ejemplos
"topic://Sport*Results"	Todos los temas que empiezan con "Sport" y acaban en "Results"	"topic://SportsResults" y "topic://Sport/Hockey/National/Div3/Results"
" topic://Deporte?Resultados "	Todos los temas que empiezan en "Sport" seguido de un carácter único, seguido de "Results"	"topic://SportsResults" y "topic://SportXResults"
"topic://Sport/*ball*/Div?/Results/*/???"	Temas	"topic://Sport/Football/Div1/Results/2002/Nov" y "topic://Sport/Netball/National/Div3/Results/02/Jan"

IBM WebSphere MQ V7.0 gestor de colas con formato comodín de nivel de tema

IBM WebSphere MQ V7.0 gestor de colas con formato comodín de nivel de tema utiliza los siguientes caracteres comodín:

para coincidir con varios niveles
+ para coincidir con un solo nivel

Tabla 2 en la página 32 proporciona algunos ejemplos sobre cómo utilizar este esquema de comodín.

Tabla 2. Ejemplos de URI en los que se utiliza el esquema de comodín a nivel de tema para el gestor de colas de IBM WebSphere MQ V7.0

Identificador uniforme de recursos	Coincide con	Ejemplos
"topic://Sport/+/Results"	Todos los temas con un nombre de nivel de jerarquía único entre Sport y Results	"topic://Sport/Football/Results" y "topic://Sport/Ju-Jitsu/Results"
"topic://Sport/#/Results"	Todos los temas que empiezan con "Sport/" y acaban en "/Results"	"topic://Sport/Football/Results" y "topic://Sport/Hockey/National/Div3/Results"
"topic://Sport/Football/#"	Todos los temas que empiezan con "Sport/Football/"	"topic://Sport/Football/Results" y "topic://Sport/Football/TeamNews/Signings/Managerial"

IBM WebSphere MQ V6.0 gestor de colas con intermediario V1

IBM WebSphere MQ V6.0 gestor de colas con el intermediario V1 utiliza los siguientes caracteres comodín:

* para 0 o más caracteres
? para 1 carácter
% para un carácter de escape

Tabla 1 en la página 32 proporciona algunos ejemplos sobre cómo utilizar este esquema de comodín.

IBM WebSphere MQ V6.0 con, o conexión en tiempo real con, un intermediario V2

IBM WebSphere MQ V6.0 con, o conexión en tiempo real con, un intermediario V2 utiliza los siguientes caracteres comodín:

para coincidir con varios niveles
+ para coincidir con un solo nivel

Tabla 2 en la página 32 proporciona algunos ejemplos sobre cómo utilizar este esquema de comodín.

WebSphere bus de integración de servicios

WebSphere bus de integración de servicios utiliza los siguientes caracteres comodín:

* para coincidir con cualquier carácter en un nivel de jerarquía
// para coincidir con 0 o más niveles
//. para coincidir con 0 o más niveles (al final de una expresión de tema)

Tabla 3 en la página 33 proporciona algunos ejemplos sobre cómo utilizar este esquema de comodín.

Tabla 3. URI de ejemplo que utilizan el esquema de comodín para WebSphere bus de integración de servicios

Identificador uniforme de recursos	**Coincide con**	**Ejemplos**
"topic://Sport/*ball/Results"	Todos los temas con un nombre de nivel jerárquico único que acaban con"ball" entre Sport y Results	"topic://Sport/Football/Results" y "topic://Sport/Netball/Results"
"topic://Sport//Results"	Todos los temas que empiezan con "Sport/" y acaban en "/Results"	"topic://Sport/Football/Results" y "topic://Sport/Hockey/National/Div3/Results"
"topic://Sport/Football//."	Todos los temas que empiezan con "Sport/Football/"	"topic://Sport/Football/Results" y "topic://Sport/Football/TeamNews/Signings/Managerial"
"topic://Sport/*ball//Results//."	Temas	"topic://Sport/Football/Results" y "topic://Sport/Netball/National/Div3/Results/2002/November"

Conceptos relacionados

Identificadores uniformes de recursos de cola
El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

Destinos temporales
Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Comodines de destino
XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Referencia relacionada

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Identificadores uniformes de recursos de cola

El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

El URI de una cola empieza con la secuencia queue://, seguida por el nombre de la cola; también podría incluir una lista de pares de nombre-valor que establecen las propiedades de cola restantes.

Para colas de IBM WebSphere MQ (pero no para colas del proveedor de mensajería predeterminado WebSphere Application Server), el gestor de colas en el cual reside la cola se puede especificar antes de la cola, con una / que separa el nombre del gestor de colas del nombre de cola.

Si se especifica un gestor de colas, debe ser uno al que XMS está conectado directamente para la conexión utilizando esta cola, o debe ser accesible desde esta cola. Los gestores de colas remotos solo están soportados para recuperar mensajes de colas, no para colocar mensajes en colas. Si desea detalles completos, consulte la documentación del gestor de colas IBM WebSphere MQ.

Si no se especifica ningún gestor de colas, el separador/ adicional es opcional, y su presencia o ausencia es indiferente para la definición de la cola.

Las definiciones de cola siguientes son todas equivalentes para una cola IBM WebSphere MQ llamada QB en un gestor de colas llamado QM_A, al que XMS está conectado directamente.

```
queue://QB
queue:///QB
queue://QM_A/QB
```

Conceptos relacionados

Identificadores uniformes de recursos de tema
El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

Destinos temporales
Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Comodines de destino
XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Referencia relacionada

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Destinos temporales

Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Normalmente, una aplicación utiliza un destino temporal para recibir respuestas a mensajes de solicitud. Para especificar el destino adónde se va a enviar un mensaje de solicitud, una aplicación llama al método Establecer JMSReplyTo del objeto Message que representa el mensaje de solicitud. El destino especificado en la llamada puede ser un destino temporal.

Aunque se utiliza una sesión para crear un destino temporal, el ámbito de un destino temporal es, realmente, la conexión que se había utilizado para crear la sesión. Cualquiera de las sesiones de la conexión puede crear productores de mensajes y consumidores de mensajes para el destino temporal. El destino temporal perdura, hasta que se suprime de forma explícita, o finaliza la conexión, lo que se produzca antes.

Cuando una aplicación crea una cola temporal, se crea una cola en el servidor de mensajería al que se conecta la aplicación. Si la aplicación está conectada a un gestor de colas, se crea una cola dinámica a partir de la cola modelo cuyo nombre se especifica mediante la propiedad XMSC_WMQ_TEMPORARY_MODEL , y el prefijo que se utiliza para formar el nombre de la cola dinámica se especifica mediante la propiedad XMSC_WMQ_TEMP_Q_PREFIX . Si la aplicación está conectada a un bus de integración de servicios, se crea una cola temporal en el bus y el prefijo que se utiliza para formar el nombre de la cola temporal se especifica mediante la propiedad XMSC_WPM_TEMP_Q_PREFIX .

Cuando una aplicación que está conectada a un bus de integración de servicios crea un tema temporal, el prefijo que se utiliza para formar el nombre del tema temporal se especifica mediante la propiedad XMSC_WPM_TEMP_TOPIC_PREFIX .

Conceptos relacionados

Identificadores uniformes de recursos de tema
El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

Identificadores uniformes de recursos de cola
El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

Comodines de destino
XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Referencia relacionada

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Comodines de destino

XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Los esquemas son:

Tipo de conexión	Esquema de comodín	Descripción
Gestor de colas IBM WebSphere MQ	*	0 o más caracteres
	?	1 carácter
	%	Carácter de escape
Conexión en tiempo real a un intermediario	#	Coincidir con varios niveles
	+	Coincidir con un solo nivel

Tipo de conexión	Esquema de comodín	Descripción
WebSphere Servicio Integration Bus	* // //.	Coincidir con cualquier carácter en un nivel de la jerarquía Coincidir con 0 o más niveles Coincidir con 0 o más niveles (al final de una expresión de tema)

Consulte también Nombres de tema y uso de caracteres comodín en expresiones de tema en la documentación de WebSphere Application Server .

Conceptos relacionados

Identificadores uniformes de recursos de tema
El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

Identificadores uniformes de recursos de cola
El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

Destinos temporales
Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Referencia relacionada

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Productores de mensajes

En XMS, se puede crear un productor de mensajes ya sea con un destino válido o sin ningún destino asociado. Al crear un productor de mensajes con un destino nulo, se debe especificar un destino válido al enviar un mensaje.

Productores de mensajes sin destino asociado

En XMS .NET, se puede crear un productor de mensajes con un destino nulo.

Para crear un productor de mensajes sin destino asociado cuando se utiliza la API .NET, se debe pasar NULL como parámetro en el método `CreateProducer()` del objeto ISession (por ejemplo, `session.CreateProducer(null)`). Sin embargo, debe especificarse un destino válido cuando se envía el mensaje.

Productores de mensajes con destino asociado

En este escenario, el productor de mensajes se crea utilizando un destino válido. Durante la operación de envío, no es necesario especificar el destino.

Consumidores de mensajes

Los consumidores de mensajes se pueden clasificar como suscriptores duraderos y suscriptores no duraderos y consumidores de mensajes síncronos y mensajes asíncronos.

Suscriptores duraderos

Un suscriptor duradero es un consumidor de mensajes que recibe todos los mensajes publicados sobre un tema, incluyendo los mensajes publicados mientras el suscriptor está inactivo.

La información de este temasección solo es pertinente si una aplicación se conecta a un gestor de colas de IBM WebSphere MQ o a un bus de integración de servicios de WebSphere. La información no es relevante para una conexión en tiempo real con un intermediario.

Para crear un suscriptor duradero para un tema, una aplicación llama al método Crear suscriptor duradero de un objeto Session, especificando como parámetros un nombre que identifica la suscripción duradera y un objeto Destination que representa el tema. La aplicación puede crear un suscriptor duradero con o sin un selector de mensajes, y puede especificar si el suscriptor duradero va a recibir mensajes publicados por su propia conexión.

La sesión utilizada para crear un suscriptor duradero debe tener un identificador de cliente asociado. El identificador de cliente es el mismo que el asociado a la conexión que se utiliza para crear la sesión; se especifica como se describe en “Objetos ConnectionFactories y Connection” en la página 24.

El nombre que identifica la suscripción duradera debe ser exclusivo dentro del identificador de cliente y, por lo tanto, el identificador de cliente forma parte del identificador único completo de la suscripción duradera. El servidor de mensajería mantiene un registro de la suscripción duradera y garantiza que se conservan todos los mensajes publicados sobre el tema, hasta que reciban un acuse de recibo del suscriptor duradero o hasta que caduquen.

El servidor de mensajería sigue manteniendo el registro de la suscripción duradera, incluso después de que se cierre el suscriptor duradero. Para reutilizar una suscripción duradera que se ha creado anteriormente, una aplicación debe crear un suscriptor duradero especificando el mismo nombre de suscripción, y utilizando una sesión con el mismo identificador de cliente, que los asociados a la suscripción duradera. Solo una sesión a la vez puede tener un suscriptor duradero para una suscripción duradera concreta.

El ámbito de una suscripción duradera es el servidor de mensajería que mantiene un registro de la suscripción. Si dos aplicaciones conectadas a distintos servidores de mensajería crean cada una un suscriptor duradero utilizando el mismo nombre de suscripción e identificador de cliente, se crean dos suscripciones duraderas completamente independientes.

Para suprimir una suscripción duradera, una aplicación llama al método Anular suscripción de un objeto Session, especificando como parámetro el nombre que identifica la suscripción duradera. El identificador de cliente asociado a la sesión debe ser el mismo que está asociado a la suscripción duradera. El servidor de mensajería suprime el registro de la suscripción duradera que está manteniendo y no envía ningún mensaje más al suscriptor duradero.

Para cambiar una suscripción existente, una aplicación puede crear un suscriptor duradero utilizando el mismo nombre de suscripción e identificador de cliente, pero especificando un tema o un selector de mensajes diferente (o ambos). El cambio de una suscripción duradera es equivalente a suprimir la suscripción y crear una nueva.

Para una aplicación que se conecta a un gestor de colas de IBM WebSphere MQ V7.0, XMS gestiona las colas de suscriptor. De ahí que no es necesario que la aplicación especifique una cola de suscriptor. XMS ignorará la cola de suscriptor, si se ha especificado.

Pero para una aplicación que se conecta a un gestor de colas de IBM WebSphere MQ V6.0, cada suscriptor duradero debe tener una cola de suscriptor designada. Para especificar el nombre de la cola de suscriptor para un tema, establezca la propiedad XMSC_WMQ_DUR_SUBQ del objeto Destination que representa el tema. La cola de suscriptor predeterminada es SYSTEM.JMS.D.SUBSCRIBER.QUEUE.

Los suscriptores duraderos que se conectan a gestores de colas de IBM WebSphere MQ V6.0 pueden compartir una misma cola de suscriptor o bien cada suscriptor duradero puede recuperar sus mensajes a partir de su propia cola de suscriptor exclusiva. Para ver un análisis sobre qué método adoptar para su aplicación, consulte *IBM WebSphere MQ - Utilización de Java*.

Tenga en cuenta que no puede cambiar la cola de suscriptor para una suscripción duradera. La única forma de cambiar la cola de suscriptor es suprimir la suscripción y crear una nueva.

Para una aplicación que se conecta a un bus de integración de servicios, cada suscriptor duradero debe tener un inicio de suscripción duradera designada. Para especificar el inicio de la suscripción duradera para todos los suscriptores duraderos que utilizan la misma conexión, establezca la propiedad XMSC_WPM_DUR_SUB_HOME del objeto ConnectionFactory que se utiliza para crear la conexión.

Para especificar el inicio de la suscripción duradera para un tema individual, establezca la propiedad XMSC_WPM_DUR_SUB_HOME del objeto Destination que representa el tema. Se debe especificar un inicio de suscripción duradera para una conexión antes de que una aplicación pueda crear una suscriptor duradero que utilice la conexión. Cualquier valor especificado para un destino altera temporalmente el valor especificado para la conexión.

Suscriptores no duraderos

Un suscriptor no duradero es un consumidor de mensajes que solo recibe mensajes que se publican mientras el suscriptor está activo. Los mensajes entregados mientras el suscriptor está inactivo se pierden.

La información de este temasección solo es pertinente si se utiliza mensajería de Publicar/Suscribir a través de un gestor de colas de IBM WebSphere MQ V6.0.

Si los objetos de consumidor no se suprimen antes o durante el cierre de la conexión, los mensajes se pueden dejar en las colas de intermediario para los suscriptores que ya no están activos.

En esta situación, las colas se pueden borrar de estos mensajes utilizando el programa de utilidad de limpieza proporcionado con IBM WebSphere MQ Classes for JMS. Los detalles sobre cómo utilizar este programa de utilidad se proporcionan en *IBM WebSphere MQ Utilización de Java.* También es posible que tenga que aumentar la profundidad de cola de la cola de suscriptor si quedan grandes números de mensajes en esta cola.

Consumidores de mensajes síncronos

El consumidor de mensajes síncronos recibe los mensajes de una cola de forma síncrona.

Un consumidor de mensajes síncrono recibe un mensaje cada vez. Cuando se utiliza el método `Receive(intervalo de espera)`; la llamada solo espera un mensaje un periodo de tiempo especificado en milisegundos, o hasta que se cierra el consumidor de mensajes.

Si se utiliza el método ReceiveNoWait(), el consumidor de mensajes síncrono recibe mensajes sin ningún retardo; si está disponible el siguiente mensaje, se recibe inmediatamente, de lo contrario, se devuelve un puntero de un objeto Message nulo.

Consumidores de mensajes asíncronos

El consumidor de mensajes asíncronos recibe mensajes de una cola de forma asíncrona. El escucha de mensajes registrado por la aplicación se invoca siempre que está disponible un nuevo mensaje en la cola.

Mensajes dañados

Un mensaje dañado es un mensaje que no puede procesar una aplicación MDB receptora. Si se encuentra un mensaje con formato incorrecto, el objeto XMS MessageConsumer puede volver a ponerlo en cola de acuerdo con dos propiedades de cola, BOQNAME y BOTHRESH.

En algunos casos, un mensaje entregado a un MDB se podría devolver a una cola de IBM WebSphere MQ. Por ejemplo, esto puede suceder si se entrega un mensaje en una unidad de trabajo que, posteriormente, se retrotrae. Un mensaje que se retrotrae, normalmente, se vuelve a entregar, pero un mensaje con un formato incorrecto podría provocar de forma repetida que falle un MDB y, por lo tanto, no se puede entregar. Dicho mensaje se denomina mensaje dañado. Puede configurar IBM WebSphere MQ de forma que el mensaje dañado se transfiere automáticamente a otra cola para una investigación adicional o para descartarse. Si desea más información sobre cómo configurar IBM WebSphere MQ de esta forma, consulte Manejo de mensajes dañados en ASF.

A veces, a una cola llega un mensaje con un formato incorrecto. En este contexto, un formato incorrecto significa que la aplicación receptora no puede procesar el mensaje correctamente. Dicho mensaje puede provocar que la aplicación receptora falle y restituya este mensaje con formato incorrecto. El mensaje se puede entregar repetidamente a la cola de entrada y la aplicación lo puede restituir también repetidamente. Estos mensajes son conocidos como mensajes dañados. El objeto XMS MessageConsumer detecta mensajes dañados y los redirecciona a un destino alternativo.

El gestor de colas IBM WebSphere MQ conserva un registro del número de veces que se ha restituido cada mensaje. Cuando este número alcanza un valor de umbral configurable, el consumidor de mensajes

vuelve a poner en cola al mensaje en una cola de retirada especificada. Si esta recolocación en cola falla por cualquier motivo, el mensaje se elimina de la cola de entrada y se vuelve a poner en cola en la cola de mensajes no entregados, o se descarta.

Los objetos XMS ConnectionConsumer manejan mensajes dañados de la misma forma y utilizan las mismas propiedades de cola. Si varios consumidores de conexiones están supervisando la misma cola, es posible que el mensaje dañado se pueda entregar a una aplicación más veces que el valor de umbral, antes de que se produzca la recolocación en cola. Este comportamiento se debe a la forma en la que los consumidores de conexionesindividuales supervisan las colas y vuelven a colocar en cola mensajes dañados.

El valor de umbral y el nombre de la cola restituida son atributos de una cola IBM WebSphere MQ. Los nombres de los atributos son BackoutThreshold y BackoutRequeueQName. La cola a la que se aplican es la siguiente:

- Para la mensajería punto a punto, esta es la cola local subyacente. Esto es importante cuando los consumidores de mensajes y los consumidores de conexiones utilizan alias de cola.
- Para la mensajería de publicación/suscripción en la modalidad normal de proveedor de mensajería IBM WebSphere MQ, esta es la cola modelo a partir de la que se ha creado la cola gestionada del objeto Tema.
- Para la mensajería de publicación/suscripción en la modalidad de migración del proveedor de mensajería IBM WebSphere MQ, esta es la cola CCSUB definida en el objeto TopicConnectionFactory, o la cola CCDSUB definida en el objeto Topic.

Para establecer los atributos BackoutThreshold y BackoutRequeueQName, emita el mandato MQSC siguiente:

```
ALTER QLOCAL(your.queue.name) BOTHRESH(threshold value)
BOQNAME(your.backout.queue.name)
```

Para la mensajería de publicación/suscripción, si el sistema crea una cola dinámica para cada suscripción, estos valores de atributo se obtienen de las clases IBM WebSphere MQ para la cola modelo JMS, SYSTEM.JMS.MODEL.QUEUE. Para modificar estos valores, utilice:

```
ALTER QMODEL(SYSTEM.JMS.MODEL.QUEUE) BOTHRESH(threshold value)
BOQNAME(your.backout.queue.name)
```

Si el valor del umbral de restitución es cero, el manejo de mensajes dañados está inhabilitado, y los mensajes dañados permanecen en la cola de entrada. De lo contrario, cuando el recuento de restitución alcanza el valor de umbral, el mensaje se envía a la cola de retirada especificada. Si el recuento de restituciones alcanza el valor umbral, pero el mensaje no puede pasar a la cola de retirada, el mensaje se envía a la cola de mensajes no entregados o se descarta. Esta situación se produce si la cola de retirada no está definida, o si el objeto MessageConsumer no puede enviar el mensaje a la cola de retirada.

Manejo de mensajes dañados en ASF
Al utilizar los Recursos del servidor de aplicaciones (ASF), el ConnectionConsumer, y no MessageConsumer, procesa mensajes dañados. El ConnectionConsumer vuelve a colocar en cola mensajes de acuerdo con las propiedades BackoutThreshold y BackoutRequeueQName de la cola.

Cuando una aplicación utiliza ConnectionConsumers, las circunstancias en las cuales se restituye un mensaje dependen de la sesión que proporciona el servidor de aplicaciones:

- Cuando la sesión es una sesión sin transacción, con AUTO_ACKNOWLEDGE o DUPS_OK_ACKNOWLEDGE, un mensaje solo se restituye después de un error del sistema, o si la aplicación termina de forma inesperada
- Cuando la sesión es una sesión sin transacción con CLIENT_ACKNOWLEDGE, los mensajes sin acuse de recibo se pueden restituir mediante el servidor de aplicaciones que llama `Session.recover()`.

  Normalmente, la implementación de cliente de MessageListener o el servidor de aplicaciones llamando a `Message.acknowledge()`. `Message.acknowledge()` reconoce todos los mensajes entregados en la sesión, hasta el momento.

- Cuando la sesión es una sesión con transacción, los mensajes no reconocidos se puede restituir mediante el servidor de aplicaciones llamando a `Session.rollback()`.

Examinadores de colas

Una aplicación utiliza un examinador de colas para examinar mensajes en una cola sin eliminarlas.

Para crear un examinador de colas, una aplicación llama al método Crear examinador de colas de un objeto ISession, especificando como parámetro un objeto Destination que identifica la cola que se va a examinar. La aplicación puede crear un examinador de colas con o sin un selector de mensajes.

Tras crear un examinador de colas, la aplicación puede llamar al método GetEnumerator del objeto IQueueBrowser para obtener una lista de los mensajes en la cola. El método devuelve un enumerador que encapsula una lista de objetos Message. El orden de los objetos Message de la lista es el mismo que el orden en el cual los mensajes deberían recuperarse de la cola. La aplicación puede utilizar el enumerador para examinar cada mensaje a su vez.

El enumerador se actualiza de forma dinámica a medida que los mensajes se colocan en la cola y se eliminan de la cola. Cada vez que la aplicación llama a IEnumerator.MoveNext() para examinar el siguiente mensaje en la cola, el mensaje refleja el contenido actual de la cola.

Una aplicación puede llamar al método GetEnumerator más de una vez para un examinador de colas determinado. Cada llamada devuelve un nuevo enumerador. Por lo tanto, la aplicación puede utilizar más de un enumerador para examinar los mensajes de una cola y mantener varias posiciones dentro de la cola.

Una aplicación puede utilizar un examinador de colas para buscar un mensaje adecuado para eliminarlo de una cola y, después, utilizar un consumidor de mensajes con un selector de mensajes para eliminar el mensaje. El selector de mensajes puede seleccionar el mensaje de acuerdo con el valor del campo de cabecera JMSMessageID. Si desea más información sobre este y otros campos de cabecera de mensaje JMS, consulte “Campos de cabecera en mensajes de Un XMS” en la página 73.

Solicitantes

Una aplicación utiliza un solicitante para enviar un mensaje de solicitud y, después, esperar y recibir la respuesta.

Muchas aplicaciones de mensajería se basan en algoritmos que envían un mensaje de solicitud y, después, esperan una respuesta. XMS proporciona una clase llamada Requestor para ayudar con el desarrollo de este estilo de aplicación.

Para crear un solicitante, una aplicación llama al constructor Create Requestor de la clase Requestor, especificando como parámetros un objeto Session y un objeto Destination que identifica dónde se van a enviar los mensajes de solicitud. La sesión no debe realizar una transacción ni tener una modalidad de acuse de recibo que sea XMSC_CLIENT_ACKNOWLEDGE. El constructor crea automáticamente una cola o tema temporal donde se van a enviar los mensajes de respuesta.

Después de crear un solicitante, la aplicación puede llamar al método Solicitar del objetoRequestor para enviar un mensaje de solicitud y, después, esperar una respuesta y recibirla de la aplicación que recibe el mensaje de solicitud. La llamada espera hasta que se recibe la respuesta o hasta que finaliza la sesión, lo que se produzca antes. El solicitante solo necesita una respuesta para cada mensaje de solicitud.

Cuando la aplicación cierra el solicitante, la cola o tema temporal se suprime. Sin embargo, la sesión asociada no se cierra.

Supresión de objeto

Cuando una aplicación suprime un objeto XMS que ha creado, XMS libera los recursos internos que se han asignado al objeto.

Cuando una aplicación crea un objeto de Un XMS, XMS asigna memoria y otros recursos internos al objeto. XMS conserva estos recursos internos hasta que la aplicación suprime de forma explícita el objeto

llamando al método cerrar o suprimir del objeto, en dicho punto XMS libera los recursos internos. Si una aplicación intenta suprimir un objeto que ya se ha suprimido, se ignora la llamada.

Cuando una aplicación suprime un objeto Connection o Session, XMS suprime determinados objetos asociados automáticamente y libera sus recursos internos. Estos son objetos que han sido creados por el objeto Connection o Session y no tienen ninguna función independiente del objeto. Estos objetos se muestra en Tabla 4 en la página 41.

Nota: Si una aplicación cierra una conexión con sesiones dependientes, todos los objetos que dependen de estas sesiones también se suprimen. Solo un objeto Connection o Session puede tener objetos dependientes.

Tabla 4. Objetos que se suprimen automáticamente

Objeto suprimido	**Método**	**Objetos dependientes que se suprimen automáticamente**
Conexión	Cerrar conexión	Objetos ConnectionMetaData y Session
Sesión (Session).	Cerrar sesión	Objetos MessageConsumer, MessageProducer, QueueBrowser y Requestor

Transacciones XA IBM WebSphere MQ gestionadas a través de XMS

Las transacciones XA IBM WebSphere MQ gestionadas se pueden utilizar mediante XMS.

Para utilizar transacciones XA a través de XMS, se debe crear una sesión con transacción. Cuando se está utilizando la transacción XA, el control de transacciones se realiza mediante las transacciones globales DTC (Distributed Transaction Coordinator) y no a través de sesiones XMS. Al utilizar transacciones XA, `Session.commit` o `Session.rollback` no se puede emitir en la sesión XMS. En su lugar, utilice los métodos `Transscope.Commit` o `Transscope.Rollback` de DTC para confirmar o retrotraer las transacciones. Si se utiliza una sesión para transacciones XA, el productor o consumidor que se crea utilizando la sesión debe formar parte de la transacción XA. No se pueden utilizar para ninguna operación fuera del ámbito de transacciones XA. No se pueden utilizar para operaciones como `Producer.send` o `Consumer.receive` fuera de la transacción XA.

Se lanza una objeto de excepción `IllegalStateException` si

- Se utiliza una sesión con transacción XA para `Session.commit` o `Session.rollback`.
- Los objetos de productor o consumidor que se han utilizado una vez en una sesión con transacción XA se utilizan fuera del ámbito de transacciones XA.

Las transacciones XA no están soportadas en consumidores asíncronos.

Nota:

1. Se podría emitir un cierre en el objeto `Producer`, `Consumer`, `Session` o `Connection` antes de confirmar la transacción XA. En cuyo caso, los mensajes de la transacción se retrotraen. De forma similar, si la conexión se interrumpe antes de confirmar la transacción XA, se retrotraen todos los mensajes de la transacción. Para un objeto `Producer` , una retrotracción significa que los mensajes no se colocan en la cola. Para un objeto `Consumer` , una retrotracción significa que los mensajes se dejan en la cola.
2. Si un objeto `Producer` coloca un mensaje con `TimeToLive` en `TransactionScope` y se emite `commit` una vez que ha transcurrido el tiempo, el mensaje puede caducar antes de que se emita `commit`. En este caso, el mensaje no se pondrá a disposición de los objetos `Consumer`.
3. Los objetos `Session` no están soportados entre las hebras. No está soportado el uso de transacciones con los objetos `Session` que se comparten entre hebras.

Tipos primitivos de XMS

XMS proporciona equivalentes de los ochos tipos primitivos de Java (byte, short, int, long, float, double, char y boolean). Esto permite el intercambio de mensajes entre XMS y JMS sin pérdida ni corrupción de datos.

Tabla 5 en la página 42 lista el tipo de datos, tamaño y valor mínimo y máximo equivalentes de Java de cada tipo primitivo de XMS.

Tabla 5. Tipos de datos de XMS y sus equivalentes Java

Tipo de datos XMS	Tipo de datos Java compatible	Tamaño	Valor mínimo	Valor máximo
System.Boolean	boolean	32 bits	falso	true
System.SBYTE	byte	8 bits	-2^7 (-128)	2^7-1 (127)
System.BYTE	byte	8 bits	-2^7 (-128)	2^7-1 (127)
System.CHAR	byte	8 bits	-2^7 (-128)	2^7-1 (127)
System.Int16	corto	16 bits	-2^{15} (-32768)	2^{15}-1 (32767)
System.Int32	int	32 bits	-2^{31} (-2147483648)	2^{31}-1 (2147483647)
System.Int64	largo	64 bits	-2^{63} (-9223372036854775808)	2^{63}-1 (9223372036854775807)
System.Single	flotante	32 bits	-3.402823E-38 (hasta 7-dígitos de precisión)	3.402823E+38 (hasta 7-dígitos de precisión)
System.Double	doble	64 bits	-1.79769313486231E-308 (hasta 15-dígitos de precisión)	1.79769313486231E+308 (hasta 15-dígitos de precisión)

Conceptos relacionados

Atributos y propiedades de objetos
Un objeto XMS puede tener atributos y propiedades, que son característicos del objeto, que se implementan de formas diferentes.

Conversión implícita de un valor de propiedad de un tipo de datos a otro
Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Conversión implícita de un valor de propiedad de un tipo de datos a otro

Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Una propiedad de un objeto tiene un nombre y un valor; el valor tiene un tipo de datos asociado, donde también se hace referencia al valor de una propiedad como *tipo de propiedad.*

Una aplicación utiliza los métodos de la clase PropertyContext para obtener y establecer las propiedades de objetos. Para poder obtener el valor de una propiedad, una aplicación llama al método que es apropiado para el tipo de propiedad. Por ejemplo, para obtener el valor de una propiedad de entero, normalmente, una aplicación llama al método GetIntProperty.

Sin embargo, cuando una aplicación obtiene el valor de una propiedad, XMS puede convertir el valor a otro tipo de datos. Por ejemplo, para obtener el valor de una propiedad de entero, una aplicación puede llamar al método GetStringProperty, que devuelve el valor de la propiedad como una serie. Las conversiones soportadas por XMS se muestran en Tabla 6 en la página 43.

Tabla 6. Conversiones soportadas de un tipo de propiedad a otros tipos de datos

Tipo de propiedad	**Tipos de datos de destino soportados**
System.String	System.Boolean, System.Double, System.Float, System.Int32, System.Int64, System.SByte, System.Int16
System.Boolean	System.String, System.SByte, System.Int32, System.Int64, System.Int16
System.Char	System.String
System.Double	System.String
System.Float	System.String, System.Double
System.Int32	System.String, System.Int64
System.Int64	System.String
System.SByte	System.String, System.Int32, System.Int64, System.Int16
System.SByte array	System.String
System.Int16	System.String, System.Int32, System.Int64

Las reglas generales siguientes rigen las conversiones soportadas:

- Los valores de propiedad numéricos se pueden convertir de un tipo de datos a otro, siempre que no se pierda ningún dato durante la conversión. Por ejemplo, el valor de una propiedad con el tipo de datos System.Int32 se puede convertir a un valor con el tipo de datos System.Int64, pero no se puede convertir a un valor con el tipo de datos System.Int16.
- Un valor de propiedad de cualquier tipo de datos se puede convertir a una serie.
- Un valor de propiedad de serie se puede convertir a cualquier otro tipo de datos, siempre que la serie tenga el formato correcto para la conversión. Si una aplicación intenta convertir un valor de propiedad de serie que no tiene un formato correcto, XMS puede devolver errores.
- Si una aplicación intenta una conversión que no está soportada, XMS puede devolver un error.

Las reglas siguientes se aplican cuando un valor de propiedad se convierte de un tipo de datos a otro:

- Al convertir una propiedad booleana a una serie, el valor verdadero se convierte a la serie "true", y el valor falso se convierte a la serie "false".
- Al convertir un valor de propiedad booleana a un tipo de datos numérico, incluido System.SByte, el valor verdadero se convierte a 1, y el valor falso se convierte a 0.
- Al convertir un valor de propiedad de serie a un valor booleano, la serie "true" (no distingue entre mayúsculas y minúsculas), o "1" se convierte a verdadero y la serie "false" (no distingue entre mayúsculas y minúsculas) o "0" se convierte a falso. Todas las demás series no se pueden convertir.
- Al convertir un valor de propiedad de serie a un valor con el tipo de datosSystem.Int32, System.Int64, System.SByte o System.Int16, la serie debe tener el formato siguiente.

  [*espacios en blanco*][*signo*]*dígitos*

  Los componentes de serie se definen del modo siguiente:

  espacios en blanco
  Caracteres en blanco iniciales opcionales.

  signo
  Un carácter opcional de signo más (+) o signo menos (-).

dígitos
Una secuencia continua de caracteres de dígitos (0-9). Al menos, debe estar presente un carácter de dígito.

Después de la secuencia de caracteres de dígitos, la serie puede contener otros caracteres que no son caracteres de dígitos, pero la conversión se detiene tan pronto como se llega al primero de estos caracteres. Se da por sentado que la serie representa un entero decimal.

XMS puede devolver un error si la serie no tiene un formato correcto.

- Al convertir un valor de propiedad de serie a un valor con el tipo de datosSystem.Double o System.Float, la serie debe tener el formato siguiente:

  [*espacios en blanco*][*signo*][*dígitos*][*punto*[*d_dígitos*]][*e_car*[*e_signo*]*e_dígitos*]

  Los componentes de serie se definen del modo siguiente:

  espacios en blanco
  (Opcional) Caracteres en blanco iniciales.

  signo
  (Opcional) Carácter de signo más (+) o signo menos (-).

  dígitos
  Una secuencia continua de caracteres de dígitos (0-9). Al menos, debe estar presente un carácter de un dígito en *dígitos* o *d_dígitos*.

  punto
  (Opcional) Punto decimal (.).

  d_dígitos
  Una secuencia continua de caracteres de dígitos (0-9). Al menos, debe estar presente un carácter de un dígito en *dígitos* o *d_dígitos*.

  e_car
  Un carácter exponente, que puede ser *E* o *e*.

  e_signo
  (Opcional) Carácter de signo más (+) o signo menos (-) para el exponente.

  e_dígitos
  Una secuencia contigua de caracteres de dígitos (0-9) para el exponente. Al menos, debe estar presente un carácter de dígito, si la serie contiene un carácter de exponente.

  Detrás de la secuencia de caracteres de dígitos, o los caracteres opcionales que representan un exponente, la serie puede contener otros caracteres que no son caracteres de dígitos, pero la conversión se detiene tan pronto como se llega al primero de estos caracteres. Se da por supuesto que la serie representa un número de coma flotante decimal con un exponente que es una potencia de 10.

  XMS puede devolver un error si la serie no tiene un formato correcto.

- Al convertir un valor de propiedad numérica a una serie, incluyendo un valor de propiedad con el tipo de datos System.SByte, el valor se convierte a la representación de serie del valor como número decimal, no la serie que contiene el carácter ASCII para dicho valor. Por ejemplo, el entero 65 se convierte a la serie "65", no la serie "A".
- Al convertir un valor de propiedad de matriz de bytes a una serie, cada byte se convierte a los 2 caracteres hexadecimales que representan el byte. Por ejemplo, la matriz de bytes {0xF1, 0x12, 0x00, 0xFF} se convierte a la serie "F11200FF".

Las conversiones de un tipo de propiedad a otros tipos de datos están soportadas por los métodos de ambas clases, Property y PropertyContext.

Conceptos relacionados

Atributos y propiedades de objetos
Un objeto XMS puede tener atributos y propiedades, que son característicos del objeto, que se implementan de formas diferentes.

Tipos primitivos de XMS
XMS proporciona equivalentes de los ochos tipos primitivos de Java (byte, short, int, long, float, double, char y boolean). Esto permite el intercambio de mensajes entre XMS y JMS sin pérdida ni corrupción de datos.

Referencia relacionada

Mensajes de correlación
El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Iteradores

Un iterador encapsula una lista de objetos y un cursor que mantiene la posición actual en la lista. El concepto de un Iterador, tal como está disponible en Message Service Client for C/C++, se implementa utilizando la interfazIEnumerator en Message Service Client for .NET.

Cuando se crea un iterador, la posición del cursor se encuentra antes del primer objeto. Una aplicación utiliza un iterador para recuperar cada objeto a su vez.

La clase Iterator de Message Service Client for C/C++ es equivalente a la clase Enumerator en Java.XMS .NET es similar a Java y utiliza una interfaz IEnumerator.

Una aplicación puede utilizar un IEnumerator para realizar las tareas siguientes:

- Para obtener las propiedades de un mensaje
- Para obtener los pares de nombre-valor del cuerpo de un mensaje de correlación
- Para examinar los mensajes de una cola
- Obtener los nombres de las propiedades de mensaje definidas por JMS que pueden ser utilizadas por una conexión

Identificadores de juego de caracteres codificado

En XMS .NET, todas las cadenas de caracteres se pasan utilizando la cadena .NET nativa. Puesto que esto tiene una codificación fija, no es necesaria más información para interpretarla. Por ello, la propiedad XMSC_CLIENT_CCSID no es necesaria para las aplicaciones XMS .NET.

Códigos de error y excepción de XMS

XMS utiliza un rango de códigos de error para indicar anomalías. Estos códigos de error no se listan de forma explícita en esta documentación porque pueden variar de release a release. Solo se documentan los códigos de excepción de XMS (con el formato XMS_X_...) porque permanecen igual entre los mismos releases de XMS.

Creación de sus propias aplicaciones

Cree sus propias aplicaciones como crea las aplicaciones de ejemplo.

Cree su aplicación .NET, tal como se describe en “Creación de aplicaciones de ejemplo de .NET” en la página 21 temasección, que también lista los requisitos previos que necesita para crear sus propias aplicaciones .NET. Si desea ayuda adicional para crear sus propias aplicaciones, utilice los archivos makefiles proporcionados para cada aplicación de ejemplo.

Consejo: Para ayudarle con el diagnóstico de problemas en el supuesto de una anomalía, es posible que encuentre útil compilar las aplicaciones con los símbolos incluidos.

Referencia relacionada

Interfaces .NET
En este temasección se documentan las interfaces de clase .NET y sus propiedades y métodos.

Propiedades de objetos XMS
Este secciónCapítulo documenta las propiedades de objeto definidas por XMS.

Reconexión automática del cliente IBM WebSphere MQ a través de XMS

Configure el cliente XMS para volver a conectarse automáticamente después de una anomalía de red, gestor de colas o servidor mientras se utiliza el cliente IBM WebSphere MQ V7.1 o superior como proveedor de mensajes.

Utilice las propiedades `WMQ_CONNECTION_NAME_LIST` y `WMQ_CLIENT_RECONNECT_OPTIONS` de la clase `MQConnectionFactory` para configurar una conexión de cliente para que se reconecte automáticamente. La reconexión automática de cliente reconecta un cliente después de una anomalía de conexión, o como opción después de detener el gestor de colas. El diseño de algunas aplicaciones cliente las invalida para la reconexión automática.

Las conexiones de cliente reconectable de forma automática pasan a ser reconectables una vez que se ha establecido la conexión.

Nota: Las propiedades `Opciones de reconexión de cliente`, `Tiempo de espera de reconexión de cliente` y `Lista de nombres de conexión` también se pueden establecer a través de la Tabla de definiciones de canal de cliente (CCDT) o habilitando la reconexión de cliente a través del archivo `mqclient.ini`.

Nota: Si las propiedades de reconexión se establecen en el objeto `ConnectionFactory` y, también, en la CCDT, la regla de prioridad es la siguiente. Si el valor predeterminado de la propiedad de lista de nombres de conexión está establecido en el objeto `ConnectionFactory`, la CCDT tiene prioridad. Si la lista de nombres de conexión no está establecida en su valor predeterminado, los valores de propiedad establecidos en el objeto `ConnectionFactory` tienen prioridad. El valor predeterminado de la lista de nombres de conexión es `localhost(1414)`.

Escritura de aplicaciones XMS para .NET

Este secciónCapítulo proporciona información para ayudarle a escribir aplicaciones XMS para .NET.

Este secciónCapítulo proporciona información específica sobre la escritura de aplicaciones XMS para .NET. Si desea información general sobre cómo escribir aplicaciones XMS, consulte “Cómo escribir aplicaciones de XMS” en la página 22.

El secciónCapítulo contiene los temassecciones siguientes:

Referencia relacionada

Interfaces .NET
En este temasección se documentan las interfaces de clase .NET y sus propiedades y métodos.

Tipos de datos para .NET

XMS .NET admite System.Boolean, System.Byte, System.SByte, System.Char, System.String, System.Single, System.Double, System.Decimal, System.Int16, System.Int32, System.Int64, System.UInt16, System.UInt32, System.UInt64 y System.Object. Los tipos de datos para XMS .NET son diferentes de los tipos de datos para XMS C++. Puede utilizar este seccióncapítulo para identificar los tipos de datos correspondientes.

La tabla siguiente muestra los tipos de datos correspondientes de XMS .NET y XMS C++ junto con una breve descripción de ellos.

Tabla 7. Tipos de datos para XMS .NET y XMS C++

Tipo de XMS .NET	**Tipo de XMS C++**	**Descripción**
System.SByte	xmsSBYTE xmsINT8	Valor de 8-bits firmado
System.Byte	xmsBYTE xmsUINT8	Valor de 8-bits sin firmar
System.Int16	xmsINT16 xmsSHORT	Valor de 16-bits firmado
System.UInt16	xmsUINT16 xmsUSHORT	Valor de 16-bits sin firmar
System.Int32	xmsINT32 xmsINT	Valor de 32-bits firmado
System.UInt32	xmsUINT32 xmsUINT	Valor de 32-bits sin firmar
System.Int64	xmsLONG xmsINT64	Valor de 64-bits firmado
System.UInt64	xmsULONG xmsUINT64	Valor de 64-bits sin firmar
System.Char	xmsCHAR16	Carácter de 16-bits sin firmar (Unicode para .NET)
System.Single	xmsFLOAT	Flotante IEEE de 32-bits
System.Double	xmsDOUBLE	Flotante IEEE de 64-bits
System.Boolean	xmsBOOL	Un valor True/False
No aplicable	xmsCHAR	Valor de 8-bits firmado o sin firmar (la firma depende de la plataforma)
System.Decimal	No aplicable	Entero firmado de 96-bits por 10^0 hasta 10^{28}
System.Object	No aplicable	Base de todos los tipos
System.String	No aplicable	Tipo de serie

Operaciones gestionadas o no gestionadas en .NET

El código gestionado se ejecuta de forma exclusiva dentro del entorno de tiempo de ejecución de lenguaje común de .NET y depende por completo de los servicios proporcionados por ese tiempo de ejecución. Una aplicación se clasifica como no gestionada si alguna parte de la aplicación se ejecuta o llama a servicios fuera del entorno de tiempo de ejecución de lenguaje común de .NET.

Actualmente, determinas funciones avanzadas no se pueden soportar dentro del entorno gestionado de .NET.

Si la aplicación requiere algunas funciones que, actualmente, no están soportadas en el entorno totalmente gestionado, puede cambiar la aplicación para utilizar el entorno no gestionado sin necesitar un cambio sustancial en la aplicación. Sin embargo, debería tener en cuenta que la pila XMS utiliza el código no gestionado cuando se realiza esta selección.

Conexiones a un IBM WebSphere MQ gestor de colas

Las conexiones gestionadas con WMQ_CM_CLIENT no soportarán conexiones SSL, comunicaciones no TCP y compresión de canal. Sin embargo, se podría dar soporte a estas conexiones mediante el uso de una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED). Si desea más información, consulte Desarrollo de aplicaciones .NET.

Si crea una fábrica de conexiones a partir de un objeto administrado en un entorno no gestionado, debe cambiar manualmente el valor para la modalidad de conexión a XMSC_WMQ_CM_CLIENT_UNMANAGED.

Conexiones a un motor de mensajería de WebSphere Servicio Integration Bus

Las conexiones a un motor de mensajería del bus de integración de servicios WebSphere que requieren el uso del protocolo SSL (incluido HTTPS) actualmente no están soportadas como código gestionado.

Referencia relacionada

XMSC_WMQ_CONNECTION_MODE

Destinos en .NET

En .NET, los destinos se crean de acuerdo con el tipo de protocolo y solo se pueden utilizar en el tipo de protocolo para el cual se han creado.

Se proporcionan dos funciones para crear destinos, una para temas y una para colas:

- `IDestination CreateTopic(String topic);`
- `IDestination CreateQueue(String queue);`

Estas funciones están disponibles en los dos objetos siguientes en la API:

- ISession
- XMSFactoryFactory

En ambos casos, estos métodos pueden aceptar una serie de estilo de URI, que puede incluir parámetros, en el formato siguiente:

```
"topic://some/topic/name?priority=5"
```

De forma alternativa, estos métodos pueden aceptar solo un nombre de destino, es decir, un nombre sin un prefijo topic:// o queue:// y sin parámetros.

Esto significa que la serie de estilo de URI siguiente:

```
CreateTopic("topic://some/topic/name");
```

generaría el mismo resultado que el nombre de destino siguiente:

```
CreateTopic("some/topic/name");
```

Con respecto a WebSphere Servicio Integration Bus JMS, los temas también se pueden especificar con un formato abreviado, que incluye tanto *topicname* como *topicspace*, pero no puede incluir parámetros:

```
CreateTopic("topicspace:topicname");
```

Propiedades en .NET

Una aplicación .NET utiliza los métodos de la interfaz PropertyContext para obtener y establecer las propiedades de objetos.

La interfaz PropertyContext encapsula métodos que obtienen y establecen propiedades. Estos métodos sonheredados, de forma directa o indirecta, por las clases siguientes:

- BytesMessage
- Conexión
- ConnectionFactory
- ConnectionMetaData
- Destination
- MapMessage
- Message
- MessageConsumer
- MessageProducer
- ObjectMessage
- QueueBrowser
- Session
- StreamMessage
- TextMessage

Si una aplicación establece el valor de una propiedad, el nuevo valor sustituye cualquier valor previo que tuviera la propiedad.

Si desea más información sobre propiedades de XMS, consulte “Propiedades de objetos XMS” en la página 183.

Para su facilidad de uso, los nombres y valores de propiedad de XMS en .NET están predefinidos como constantes públicos en una estructura llamada XMSC. Los nombres de estas constantes tienen el formato XMSC.< *constant*>; por ejemplo, XMSC.USERID (una constante de nombre de propiedad) y XMSC.DELIVERY_AS_APP (una constante de valor).

Además, puede acceder a constantes de IBM WebSphere MQ utilizando la estructura IBM.XMS.MQC. Si IBM.XMS ya está importado, puede acceder a los valores de estas propiedades con el formato MQC.< *constant*>. Por ejemplo, MQC.MQRO_COA_WITH_FULL_DATA.

Además, si tiene una aplicación híbrida que utiliza ambas clases, XMS .NET y IBM WebSphere MQ, para .NET y que importa ambos espacios de nombres, IBM.XMS e IBM.WMQ, debe cualificar por completo el espacio de nombres de estructura MQC para garantizar que cada aparición es única.

Actualmente, algunas funciones no está soportadas en el entorno .NET gestionado. Consulte “Operaciones gestionadas o no gestionadas en .NET” en la página 48 si desea más detalles.

Manejo de propiedades no existentes en .NET

El manejo de propiedades no existentes en XMS .NET es ampliamente compatible con la especificación JMS y, también, con las implementaciones C y C++ de XMS.

En JMS, el acceso a una propiedad no existente puede generar una excepción del sistema Java cuando un método intenta convertir el valor no existente (nulo) al tipo necesario. Si una propiedad no existe, se producen las excepciones siguientes:

- getStringProperty y getObjectProperty devuelven null
- getBooleanProperty devuelve false porque Boolean.valueOf(null) devuelve false
- getIntProperty etc.throw java.lang.NumberFormatException porque Integer.valueOf(null) lanza la excepción

Si una propiedad no existe en XMS .NET, se producen las excepciones siguientes:

- GetStringProperty y GetObjectProperty (y GetBytesProperty) devuelven null (que es el mismo que Java)
- GetBooleanProperty lanza System.NullReferenceException
- GetIntProperty etc. lanza System.NullReferenceException

Esta implementación es diferente de Java, pero es ampliamente compatible con la especificación JMS y con las interfaces C y C++ de XMS. Al igual que la implementación de Java, XMS .NET propaga cualquier excepción desde la llamada System.Convert al interlocutor. Sin embargo, a diferencia de Java, XMS lanza de forma explícita NullReferenceExceptions, en lugar de solo utilizar el comportamiento nativo de la infraestructura de .NET pasando un nulo a las rutinas de conversión del sistema.
Si la aplicación establece una propiedad en una serie como "abc" y llama a GetIntProperty, la excepción System.FormatException lanzada por Convert.ToInt32("abc") se propaga al interlocutor, que es compatible con Java. MessageFormatException solo se lanza si los tipos utilizados para setProperty y getProperty son incompatibles. Este comportamiento también es compatible con Java.

Manejo de errores en .NET

Todas las excepciones de XMS .NET se han derivado de System.Exception. Las llamadas del método XMS pueden lanzar excepciones específicas de XMS como, por ejemplo, MessageFormatException, excepciones generales XMSExceptions o excepciones del sistema como NullReferenceException.

Escriba aplicaciones para capturar cualquiera de estos errores, ya sea en bloques catch específicos o en bloques catch `System.Exception` generales, según sea lo más apropiado para los requisitos de las aplicaciones.

Escuchas de mensajes y excepción en .NET

Una aplicación .NET utiliza un escucha de mensajes para recibir mensajes de forma asíncrona, y utiliza un escucha de excepción para que se le notifique un problema con una conexión de forma asíncrona.

La funcionalidad de los escuchas de mensajes y de excepciones es la misma para .NET y para C++. Sin embargo, existen algunas pequeñas diferencias de implementación.

Escuchas de mensajes en .NET

Para recibir mensajes de forma asíncrona, debe completar los pasos siguientes:

1. Definir un método que coincida con la firma delegada del escucha de mensajes. El método que defina puede ser estático o un método de instancia y se puede definir en cualquier clase accesible. La firma delegada es la siguiente:

   ```
   public delegate void MessageListener(IMessage msg);
   ```

   y, por lo tanto, podría definir el método como:

   ```
   void SomeMethodName(IMessage msg);
   ```

2. Crear una instancia de este método como delegado utilizando algo similar a lo siguiente:

   ```
   MessageListener OnMsgMethod = new MessageListener(SomeMethodName)
   ```

3. Registrar el delegado con uno o más consumidores estableciéndolo en la propiedad MessageListener del consumidor

   ```
   consumer.MessageListener = OnMsgMethod;
   ```

Puede eliminar el delegado volviendo a establecer MessageListener en null:

```
consumer.MessageListener = null;
```

Escuchas de excepciones en .NET

El escucha de excepción funciona de una forma muy similar que el escucha de mensajes, pero tiene una definición de delegado diferente y se asigna a la conexión, en lugar de al consumidor de mensajes. Esto es igual que para C++.

1. Defina el método. La firma delegada es la siguiente:

   ```
   public delegate void ExceptionListener(Exception ex);
   ```

   y, por lo tanto, el método definido podría ser:

   ```
   void SomeMethodName(Exception ex);
   ```

2. Cree una instancia de este método como delegado utilizando algo similar a:

   ```
   ExceptionListener OnExMethod = new ExceptionListener(SomeMethodName)
   ```

3. Registre el delegado con la conexión estableciendo su propiedad ExceptionListener:

   ```
   connection.ExceptionListener = OnExMethod ;
   ```

   Puede eliminar el delegado restableciendo ExceptionListener en:

   ```
   null:  connection.ExceptionListener = null;
   ```

Cuando no se conserva ninguna referencia a los mismos, el recopilador de basura de los sistemas suprime las excepciones o los mensajes automáticamente.

A continuación, se muestra un código de ejemplo:

```
using System;
using System.Threading;
using IBM.XMS;

public class Sample
{
    public static void Main()
    {
        XMSFactoryFactory factoryFactory = XMSFactoryFactory.GetInstance(XMSC.CT_RTT);

        IConnectionFactory connectionFactory = factoryFactory.CreateConnectionFactory();
        connectionFactory.SetStringProperty(XMSC.RTT_HOST_NAME, "localhost");
        connectionFactory.SetStringProperty(XMSC.RTT_PORT, "1506");

        //
        // Create the connection and register an exception listener
        //

        IConnection connection = connectionFactory.CreateConnection();
        connection.ExceptionListener = new ExceptionListener(Sample.OnException);

        ISession session = connection.CreateSession(false, AcknowledgeMode.AutoAcknowledge);
        IDestination topic = session.CreateTopic("topic://xms/sample");

        //
        // Create the consumer and register an async message listener
        //

        IMessageConsumer consumer = session.CreateConsumer(topic);
        consumer.MessageListener = new MessageListener(Sample.OnMessage);

        connection.Start();

        while (true)
        {
            Console.WriteLine("Waiting for messages....");
```

```
            Thread.Sleep(1000);
        }
    }

    static void OnMessage(IMessage msg)
    {
        Console.WriteLine(msg);
    }

    static void OnException(Exception ex)
    {
        Console.WriteLine(ex);
    }
}
```

Cómo trabajar con objetos administrados

Este secciónCapítulo proporciona información sobre los objetos administrados. Las aplicaciones XMS pueden recuperar definiciones de objeto de un repositorio central de objetos administrados y utilizarlas para crear fábricas de conexiones y destinos.

Este secciónCapítulo proporciona información para ayudarle a crear y gestionar objetos administrados, y describe los tipos de repositorio de objetos administrados que se pueden utilizar con XMS. El secciónCapítulo también explica cómo una aplicación XMS establece una conexión con un repositorio de objetos administrados para recuperar los objetos administrados necesarios.

El secciónCapítulo contiene los temassecciones siguientes:

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Tipos soportados de repositorio de objetos administrados

XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Los directorios de objetos del sistema de archivos adoptan la forma de objetos JNDI (Java y Naming Directory Interface) serializados. Los directorios de objetos LDAP son directorios que contienen objetos JNDI. Los directorios de objetos del sistema de archivos y LDAP se pueden administrar utilizando la herramienta JMSAdmin, que se proporciona con IBM WebSphere MQ v6.0, o WebSphere MQ Explorer, que se proporciona con WebSphere MQ v7.0 y posterior. Tanto el sistema de archivos como los directorios de objetos LDAP se pueden utilizar para administrar conexiones de cliente centralizando fábricas de conexiones y destinos de IBM WebSphere MQ . El administrador de red puede desplegar varias aplicaciones que hacen referencia al mismo repositorio central y que se actualizan de forma automática para reflejar los cambios en los valores de conexión realizados en el repositorio central.

Un directorio de denominación COS contiene fábricas de conexiones y destinos WebSphere Servicio Integration Bus y se puede administrar utilizando la consola de administración de WebSphere Application Server. Para que una aplicación XMS recupere objetos del directorio de denominación COS, se debe desplegar un servicio web de búsqueda JNDI. Este servicio web no está disponible en todos los WebSphere tecnologías de integración de servicios. Consulte la documentación del producto para obtener detalles.

Nota: Reinicie las conexiones de aplicaciones para que entren en vigor los cambios en el directorio de objeto.

Conceptos relacionados

Correlación de propiedades para objetos administrados

Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Objetos administrados
Mediante el uso de objetos administrados, puede administrar los valores de la conexión utilizada por aplicaciones cliente que se van a administrar desde un repositorio central. Una aplicación recupera definiciones de objeto del repositorio central y las utiliza para crear objetos `ConnectionFactory` y `Destination`. Utilizando objetos administrados, puede desacoplar aplicaciones de los recursos que utilizan durante el tiempo de ejecución.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

Correlación de propiedades para objetos administrados

Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Para poder crear, por ejemplo, una fábrica de conexiones XMS con propiedades recuperadas de una fábrica de conexiones JMS de IBM WebSphere MQ, las propiedades deben estar correlacionadas entre las dos.

Todas las correlaciones de propiedad se realizan automáticamente.

La tabla siguiente demuestra las correlaciones entre algunas de las propiedades más comunes de fábricas de conexiones y destinos. Las propiedades que se muestran en esta tabla solo son un pequeño conjunto de ejemplos, y no todas las propiedades mostradas son relevantes para todos los tipos de conexión y servidores.

Tabla 8. Ejemplos de correlación de nombres para propiedades de fábrica de conexiones y destino

IBM WebSphere MQ Nombre de la propiedad JMS	**Nombre de propiedad XMS**
PERSISTENCE (PER)	XMSC_DELIVERY_MODE
EXPIRY (EXP)	XMSC_TIME_TO_LIVE
PRIORITY (PRI)	XMSC_PRIORITY

Tabla 9. Ejemplos de correlación de nombres para propiedades de fábrica de conexiones y destino

IBM WebSphere MQ Nombre de la propiedad JMS	**Nombre de propiedad XMS**	**Nombre de propiedad WebSphere Servicio Integration Bus**
PERSISTENCE (PER)	XMSC_DELIVERY_MODE	
EXPIRY (EXP)	XMSC_TIME_TO_LIVE	
PRIORITY (PRI)	XMSC_PRIORITY	
	XMSC_WPM_HOST_NAME	serverName
	XMSC_WPM_BUS_NAME	busName
	XMSC_WPM_TOPIC_SPACE	topicName

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

Propiedades necesarias para objetos ConnectionFactory administrados

Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Las propiedades listadas en las tablas siguientes son el mínimo necesario para establecer para una aplicación para crear una conexión a un servidor de mensajería. Si desea personalizar la forma en la que se crea una conexión, la aplicación puede establecer cualquier propiedad adicional del objeto ConnectionFactory, según sea necesario. Si desea más información, consulte “Propiedades de ConnectionFactory” en la página 185. Se incluye una lista completa de propiedades disponibles.

Conexión a un gestor de colas de IBM WebSphere MQ

Tabla 10. Valores de propiedad para objetos ConnectionFactory administrados para conexiones a un gestor de colas IBM WebSphere MQ

XMS necesario	**Propiedad JMS IBM WebSphere MQ equivalente necesaria**
XMSC_CONNECTION_TYPE	XMS resuelve el nombre de clase de la fábrica de conexiones y la propiedad TRANSPORT (TRAN).
XMSC_WMQ_HOST_NAME	HOSTNAME (HOST)
XMSC_WMQ_PORT	PORT
XMSC_WMQ_QUEUE_MANAGER	Nombre del gestor de colas

Conexión en tiempo real a un intermediario

Tabla 11. Valores de propiedad para objetos ConnectionFactory administrados para conexiones en tiempo real a un intermediario

XMS necesario	Propiedad JMS IBM WebSphere MQ equivalente necesaria
XMSC_CONNECTION_TYPE	XMS resuelve el nombre de clase de la fábrica de conexiones y la propiedad TRANSPORT (TRAN).
XMSC_RTT_HOST_NAME	HOSTNAME (HOST)
XMSC_RTT_PORT	PORT

Conexión a un WebSphere Servicio Integration Bus

Tabla 12. Valores de propiedad para objetos ConnectionFactory administrados para conexiones a un WebSphere Servicio Integration Bus

Propiedad XMS	Descripción
XMSC_CONNECTION_TYPE	El tipo del servidor de mensajería al que se conecta una aplicación.. Esto se determina a partir del nombre de clase de fábrica de conexiones.
XMSC_WPM_BUS_NAME	Para una fábrica de conexiones, el nombre del bus de integración de servicios al que se conecta la aplicación o, para un destino, el nombre del bus de integración de servicios en el cual existe el destino.

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Conexiones seguras a un IBM WebSphere MQ gestor de colas
Para habilitar una aplicación XMS .NET para realizar conexiones seguras con un IBM WebSphere MQ gestor de colas, las propiedades relevantes deben estar definidas en el objeto ConnectionFactory .

Conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus
Para permitir que una aplicación XMS de .NET establezca conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus, se deben definir las propiedades pertinentes en el objeto ConnectionFactory.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

Propiedades necesarias para objetos Destination administrados

Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

Tabla 13. Valores de propiedad de JMS IBM WebSphere MQ para objetos Destination administrados

Tipo de conexión	Propiedad	Descripción
IBM WebSphere MQ gestor de colas	QUEUE (QU) TOPIC (TOP)	La cola a la que desea conectarse El tema que utiliza la aplicación como destino
Conexión en tiempo real a un intermediario	TOPIC (TOP)	El tema que utiliza la aplicación como destino

Tabla 14. Valores de propiedad para objetos Destination administrados

Tipo de conexión	Propiedad	Descripción
IBM WebSphere MQ gestor de colas	QUEUE (QU) TOPIC (TOP)	La cola a la que desea conectarse El tema que utiliza la aplicación como destino
Conexión en tiempo real a un intermediario	TOPIC (TOP)	El tema que utiliza la aplicación como destino
WebSphere Servicio Integration Bus	topicName queueName	Si la aplicación está conectándose a un tema Si la aplicación se está conectando a una cola

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Creación de objetos administrados

Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Antes de empezar

Si desea más detalles sobre los distintos tipos del repositorio de objetos administrados que admite XMS, consulte “Tipos soportados de repositorio de objetos administrados” en la página 52.

Acerca de esta tarea

Para crear objetos administrados para IBM WebSphere MQ, utilice IBM WebSphere MQ Explorer o la herramienta de administración JMSAdmin de IBM WebSphere MQ.

Para crear los objetos administrados para IBM WebSphere MQ, WebSphere Event Brokero WebSphere Message Broker, utilice la herramienta de administración JMS de IBM WebSphere MQ (JMSAdmin).

Para crear objetos administrados para WebSphere Servicio Integration Bus, utilice la consola de administración de WebSphere Application Server.

Los pasos siguientes resumen lo qué se debe hacer para crear objetos administrados.

Procedimiento

1. Crear una fábrica de conexiones y definir las propiedades necesarias para crear una conexión de la aplicación al servidor elegido.

   Las propiedades mínimas que requiere XMS para realizar una conexión están definidas en “Propiedades necesarias para objetos ConnectionFactory administrados” en la página 55.
2. Crear el destino necesario en el servidor de mensajería, al que se conecta la aplicación:
   - Para una conexión con un IBM WebSphere MQ gestor de colas, cree una cola o un tema.
   - Para una conexión en tiempo real a un intermediario, cree un tema.
   - Para una conexión con un WebSphere Servicio Integration Bus, cree una cola o un tema.

   Las propiedades mínimas que requiere XMS para realizar una conexión están definidas en “Propiedades necesarias para objetos Destination administrados” en la página 57.

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Objetos administrados
Mediante el uso de objetos administrados, puede administrar los valores de la conexión utilizada por aplicaciones cliente que se van a administrar desde un repositorio central. Una aplicación recupera definiciones de objeto del repositorio central y las utiliza para crear objetos `ConnectionFactory` y `Destination`. Utilizando objetos administrados, puede desacoplar aplicaciones de los recursos que utilizan durante el tiempo de ejecución.

Cómo trabajar con objetos administrados
Este secciónCapítulo proporciona información sobre los objetos administrados. Las aplicaciones XMS pueden recuperar definiciones de objeto de un repositorio central de objetos administrados y utilizarlas para crear fábricas de conexiones y destinos.

Objetos ConnectionFactories y Connection
Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Tareas relacionadas

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

IDestination (para la interfaz .NET)
Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Propiedades de Destination
Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Objetos InitialContext

Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Acerca de esta tarea

Un objeto InitialContext encapsula una conexión con el repositorio. La API XMS proporciona métodos para realizar las tareas siguientes:

- Crear un objeto InitialContext

- Busca un objeto administrado en el repositorio de objetos administrados.

Si desea más detalles sobre cómo crear un objetoInitialContext, consulte “InitialContext” en la página 114 for .NET y “Propiedades de InitialContext” en la página 194.

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

InitialContext (para la interfaz .NET)
Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.

Propiedades de InitialContext
Una descripción general de las propiedades del objeto InitialContext, con enlaces a información de referencia más detallada.

Propiedades InitialContext

Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

En JNDI y en la implementación de .NET de XMS, la información adicional se proporciona en un entorno de tabla hash al constructor.

La ubicación del repositorio de objetos administrados se define en la propiedad XMSC_IC_URL. Esta propiedad normalmente se pasa en la llamada Create, pero se puede modificar para conectarse a un directorio de denominación diferente antes de la búsqueda. Para contextos de FileSystem o LDAP, esta propiedad define la dirección del directorio. Para la denominación COS, esta es la dirección del servicio web que utiliza estas propiedades para conectarse al directorio JNDI.

Las propiedades siguientes se pasan sin modificar al servicio Web, el cual las utilizará para conectar con el directorio JNDI.

- XMSC_IC_PROVIDER_URL
- XMSC_IC_SECURITY_CREDENTIALS
- XMSC_IC_SECURITY_AUTHENTICATION
- XMSC_IC_SECURITY_PRINCIPAL
- XMSC_IC_SECURITY_PROTOCOL

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

InitialContext (para la interfaz .NET)
Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.

Propiedades de InitialContext
Una descripción general de las propiedades del objeto InitialContext, con enlaces a información de referencia más detallada.

Formato URI para contextos iniciales de XMS

La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Contexto FileSystem

Para el contexto FileSystem, el URL proporciona la ubicación del directorio basado en el sistema de archivos. La estructura del URL se define mediante RFC 1738, *Localizadores uniformes de recursos (URL)*: el URL tiene el prefijo `file://` y la sintaxis que aparece detrás de este prefijo es una definición válida de un archivo que se puede abrir en el sistema en el cual se está ejecutando XMS.

Esta sintaxis puede ser específica de la plataforma y puede utilizar separadores '/ o separadores '\'. Si utiliza '\', se debe añadir a cada separador un carácter de escape utilizando un '\' adicional. Esto evita que la infraestructura .NET intente interpretar el separador como un carácter de escape para lo que aparece detrás.

Estos ejemplos ilustran esta sintaxis:

```
file://myBindings
file:///admin/.bindings
file://\\admin\\.bindings
file://c:/admin/.bindings
file://c:\\admin\\.bindings
file://\\\\madison\\shared\\admin\\.bindings
file:///usr/admin/.bindings
```

Contexto LDAP

Para el contexto LDAP, la estructura básica del URL se define mediante RFC 2255, *El formato del URL deLDAP*, que el prefijo `ldap://` que no distingue entre mayúsculas y minúsculas

La sintaxis precisa se ilustra en el ejemplo siguiente:

```
LDAP://[Hostname][:Port]["/"[DistinguishedName]]
```

Esta sintaxis es la que se define en RFC, pero sin el soporte para ningún atributo, ámbito, filtro o extensión.

Entre los ejemplos de esta sintaxis se incluye:

```
ldap://madison:389/cn=JMSData,dc=IBM,dc=UK
ldap://madison/cn=JMSData,dc=IBM,dc=UK
LDAP:///cn=JMSData,dc=IBM,dc=UK
```

Contexto WSS

Para el contexto WSS, el URL tiene el formato de un punto final de servicios web, con el prefijo `http://`.

De forma alternativa, puede utilizar el prefijo `cosnaming://` o `wsvc://`,

Estos dos prefijos se interpretan como que utiliza un contexto WSS con el URL al que se accede a través de http, lo que permite que se derive el tipo de contexto inicial fácil y directamente desde el URL.

Entre los ejemplos de esta sintaxis se incluye lo siguiente:

```
http://madison.ibm.com:9080/xmsjndi/services/JndiLookup
cosnaming://madison/jndilookup
```

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Servicio web de búsqueda JNDI
Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

InitialContext (para la interfaz .NET)

Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.

Propiedades de InitialContext
Una descripción general de las propiedades del objeto InitialContext, con enlaces a información de referencia más detallada.

Servicio web de búsqueda JNDI

Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

El servicio web se proporciona en el archivo EAR (Enterprise Archive) SIBXJndiLookupEAR.ear, situado en el directorio de instalación. Para el release actual de Message Service Client for .NET, SIBXJndiLookupEAR.ear se puede encontrar en el directorio `<install_dir>\java\lib` . Esto se puede instalar en un servidor de bus de integración de servicios de WebSphere utilizando la consola administrativa o la herramienta de scripts wsaadmin. Consulte la documentación del producto para obtener información adicional sobre el despliegue de aplicaciones de servicio web.

Para definir el servicio web en aplicaciones XMS, simplemente tendrá que establecer la propiedad XMSC_IC_URL del objeto InitialContext en el URL de punto final de servicio web. Por ejemplo, si el servicio web se despliega en un host de servidor denominado MyHost, esto sería un ejemplo de URL de punto final de servicio web:

```
wsvc://MyHost:9080/SIBXJndiLookup/services/JndiLookup
```

Establecer la propiedad XMSC_IC_URL permite que las llamadas de búsqueda de InitialContext puedan invocar el servicio web en el punto final definido, que a su vez busca el objeto administrado necesario en el servicio de denominación COS.

Las aplicaciones .NET pueden utilizar el servicio web. El despliegue en el extremo servidor es el mismo para XMS C, /C++ y XMS .NET.XMS.NET invoca el servicio web directamente a través de la infraestructura de Microsoft .NET .

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Instalación de Message Service Client for .NET mediante el asistente de instalación
La instalación utiliza un instalador InstallShield X/Windows MSI. Están disponibles dos opciones de configuración, de forma que pueda elegir entre una instalación completa o personalizada.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

Recuperación de objetos administrados

XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Los objetos que se van a recuperar pueden tener los tipos de nombres siguientes:

- Un nombre sencillo que describe el objeto Destination, por ejemplo, un destino de cola llamado SalesOrders
- Un nombre compuesto, que puede estar formado de SubContexts, separados por '/', y debe acabar con el nombre de objeto. Un ejemplo de un nombre compuesto es "Warehouse/PickLists/DispatchQueue2" donde Warehouse y Picklists son SubContexts en el directorio de denominación, y DispatchQueue2 es el nombre de un objeto Destination.

Conceptos relacionados

Tipos soportados de repositorio de objetos administrados
XMS da soporte a tres tipos de directorios de objetos administrados: Sistema de archivos, LDAP (Lightweight Directory Access Protocol) y denominación COS. Los objetos administrados del sistema de archivo y LDAP se pueden utilizar para conectarse a IBM WebSphere MQ y WebSphere Application Server, mientras que la Denominación COS solo se puede utilizar para conectarse al WebSphere Application Server.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Servicio web de búsqueda JNDI

Para acceder a un directorio de denominación COS desde XMS, se debe desplegar un servicio web de búsqueda JNDI en un servidor WebSphere Servicio Integration Bus. Este servicio web convierte la información Java del servicio de denominación COS en un formulario que las aplicaciones XMS pueden leer.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

InitialContext (para la interfaz .NET)
Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.

Propiedades de InitialContext
Una descripción general de las propiedades del objeto InitialContext, con enlaces a información de referencia más detallada.

Protección de comunicaciones para aplicaciones XMS

Este secciónCapítulo proporciona información sobre cómo configurar comunicaciones seguras para permitir que las aplicaciones XMS se conecten a través de SSL (Secure Sockets Layer) a un motor de mensajería de WebSphere Servicio Integration Bus o IBM WebSphere MQ gestor de colas.

Este secciónCapítulo proporciona información sobre cómo configurar las propiedades de ConnectionFactory de XMS para permitir que las aplicaciones establezcan conexiones seguras.

El secciónCapítulo contiene los temassecciones siguientes:

Conexiones seguras a un IBM WebSphere MQ gestor de colas

Para habilitar una aplicación XMS .NET para realizar conexiones seguras con un IBM WebSphere MQ gestor de colas, las propiedades relevantes deben estar definidas en el objeto ConnectionFactory .

El protocolo utilizado en la negociación de cifrado puede ser Secure Sockets Layer (SSL) o Transport Layer Security (TLS), en función de la CipherSuite que especifique en el objeto ConnectionFactory.

Si utiliza las bibliotecas de cliente de IBM WebSphere MQ Versión 7.0.0.1 y posteriores y se conecta a un gestor de colas de IBM WebSphere MQ Versión 7, puede crear varias conexiones con el mismo gestor de colas en la aplicación XMS . Sin embargo, no está permitida una conexión a un gestor de colas diferente. Si lo intenta, obtendrá el error `MQRC_SSL_ALREADY_INITIALIZED`.

Si utiliza las bibliotecas de cliente de IBM WebSphere MQ Versión 6 y posteriores, puede crear una conexión SSL sólo si cierra primero cualquier conexión SSL anterior. No están permitidas varias conexiones SSL simultáneas del mismo proceso a los mismos gestores de colas o gestores de colas diferentes. Si intenta más de una solicitud, obtendrá el aviso `MQRC_SSL_ALREADY_INITIALIZED`, lo que podría significar que se han ignorado algunos de los parámetros solicitados para la conexión SSL.

La tabla siguiente muestra las propiedades de ConnectionFactory para conexiones establecidas mediante SSL con un gestor de colas de IBM WebSphere MQ, acompañadas de una breve descripción:

Tabla 15. Propiedades de ConnectionFactory para conexiones a un IBM WebSphere MQ gestor de colas a través de SSL	
Nombre de la propiedad	**Descripción**
XMSC_WMQ_SSL_CERT_STORES	Las ubicaciones de los servidores que contienen las listas de revocaciones de certificados (CRL) que se van a utilizar en una conexión SSL a un gestor de colas.
XMSC_WMQ_SSL_CIPHER_SPEC	El nombre de la CipherSpec que se va a utilizar en una conexión segura a un gestor de colas.
XMSC_WMQ_SSL_CIPHER_SUITE	El nombre de la CipherSuite que se va a utilizar en una conexión SSL o TLS con un gestor de colas. El protocolo utilizado en la negociación de la conexión segura depende de la CipherSuite especificada.
XMSC_WMQ_SSL_CRYPTO_HW	Detalles de configuración para el hardware de cifrado conectado al sistema cliente.
XMSC_WMQ_SSL_FIPS_REQUIRED	El valor de esta propiedad determina si una aplicación puede o no puede utilizar suites de cifrado no compatibles con FIPS. Si esta propiedad se establece en true (verdadero), solo se utilizan algoritmos FIPS para la conexión cliente/servidor.
XMSC_WMQ_SSL_KEY_REPOSITORY	La ubicación de un archivo de base de datos de claves en el cual se almacenan claves y certificados.
XMSC_WMQ_SSL_KEY_RESETCOUNT	El KeyResetCount representa el número total de bytes sin cifrado enviados y recibidos en una conversación SSL antes de renegociar la clave secreta.
XMSC_WMQ_SSL_PEER_NAME	El nombre de igual que se va a utilizar en una conexión SSL a un gestor de colas.

Referencia relacionada

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Correlaciones de nombres CipherSuite y CipherSpec para conexiones a un IBM WebSphere MQgestor de colas

La propiedad InitialContext convierte entre la propiedad de fábrica de conexiones JMSAdmin SSLCIPHERSUITE y la propiedad XMSC_WMQ_SSL_CIPHER_SPEC de XMS casi equivalente. Es necesaria una conversión similar si especifica un valor para XMSC_WMQ_SSL_CIPHER_SUITE pero omite el valor paraXMSC_WMQ_SSL_CIPHER_SPEC.

Tabla 16 en la página 68 lista las CipherSpecs disponibles y sus equivalentes JSSE CipherSuite .

Tabla 16. CipherSpecs disponibles y sus equivalentes de JSSE CipherSuite	
CipherSpec	**JSSE CipherSuite equivalente**
DES_SHA_EXPORT	SSL_RSA_WITH_DES_CBC_SHA

Tabla 16. CipherSpecs disponibles y sus equivalentes de JSSE CipherSuite (continuación)	
CipherSpec	**JSSE CipherSuite equivalente**
DES_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
FIPS_WITH_3DES_EDE_CBC_SHA	SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
FIPS_WITH_DES_CBC_SHA	SSL_RSA_FIPS_WITH_DES_CBC_SHA
NULL_MD5	SSL_RSA_WITH_NULL_MD5
NULL_SHA	SSL_RSA_WITH_NULL_SHA
RC2_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
RC4_56_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
RC4_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC4_40_MD5
RC4_MD5_US	SSL_RSA_WITH_RC4_128_MD5
RC4_SHA_US	SSL_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	SSL_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA	SSL_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA	SSL_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA	SSL_RSA_WITH_DES_CBC_SHA
TRIPLE_DES_SHA_US	SSL_RSA_WITH_3DES_EDE_CBC_SHA

Nota: La correlación uno a uno para el nombre de paquete de cifrado SSL_RSA_WITH_3DES_EDE_CBC_SHA o SSL_RSA_WITH_DES_CBC_SHA debe tener en cuenta el valor de la propiedad XMSC_WMQ_SSL_FIPSREQUIRED y aplicar un valor heurístico.

Si especifica SSL_RSA_WITH_3DES_EDE_CBC_SHA o SSL_RSA_WITH_DES_CBC_SHA para la propiedad XMSC_WMQ_SSL_CIPHER_SUITE, y no hay ningún valor para XMSC_WMQ_SSL_CIPHER_SPEC, se elige un valor para XMSC_WMQ_SSL_CIPHER_SPEC de acuerdo con las siguientes tablas.

Los valores utilizados para XMSC_WMQ_SSL_CIPHER_SPEC cuando especifica SSL_RSA_WITH_3DES_EDE_CBC_SHA para la propiedad XMSC_WMQ_SSL_CIPHER_SUITE se muestran en la tabla siguiente:

Tabla 17. Valores utilizados para XMSC_WMQ_SSL_CIPHER_SPEC cuando especifica SSL_RSA_WITH_3DES_EDE_CBC_SHA para la propiedad XMSC_WMQ_SSL_CIPHER_SUITE	
Entrada: valor de XMSC_WMQ_SSL_FIPSREQUIRED	**Salida: XMSC_WMQ_SSL_CIPHER_SPEC elegido**
false (es decir, MQSSL_FIPS_NO)	TRIPLE_DES_SHA_US
true (es decir, MQSSL_FIPS_YES)	TLS_RSA_WITH_3DES_EDE_CBC_SHA

Nota:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA está en desuso. Sin embargo, se sigue pudiendo utilizar para transferir hasta 32 GB de datos antes de que se termine la conexión con el error AMQ9288. Para evitar este error, tendrá que evitar utilizar el triple DES, o habilitar el restablecimiento de clave secreta cuando se utiliza esta CipherSpec.

Los valores utilizados para XMSC_WMQ_SSL_CIPHER_SPEC cuando especifica SSL_RSA_WITH_DES_CBC_SHA para la propiedad XMSC_WMQ_SSL_CIPHER_SUITE se muestran en la tabla siguiente:

Tabla 18. Valores utilizados para XMSC_WMQ_SSL_CIPHER_SPEC cuando especifica SSL_RSA_WITH_DES_CBC_SHA para la propiedad XMSC_WMQ_SSL_CIPHER_SUITE	
Entrada: valor de XMSC_WMQ_SSL_FIPSREQUIRED	**Salida: XMSC_WMQ_SSL_CIPHER_SPEC elegido**
false (es decir, MQSSL_FIPS_NO)	DES_SHA_EXPORT
true (es decir, MQSSL_FIPS_YES)	TLS_RSA_WITH_DES_CBC_SHA

Conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus

Para permitir que una aplicación XMS de .NET establezca conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus, se deben definir las propiedades pertinentes en el objeto ConnectionFactory.

XMS proporciona soporte SSL y HTTPS para conexiones a un WebSphere Servicio Integration Bus. SSL y HTTPS proporcionan conexiones seguras para la autenticación y la confidencialidad.

Al igual que la seguridad WebSphere, la seguridad XMS se configura con respecto a los estándares de seguridad y convenios de denominación JSSE, que incluyen el uso de CipherSuites para especificar los algoritmos que se utilizan cuando se negocia una conexión segura. El protocolo utilizado en la negociación de cifrado puede ser SSL o TLS, en función de la CipherSuite que especifique en el objeto ConnectionFactory.

Nota: Para una aplicación .NET, las funciones de seguridad son proporcionadas por Microsoft Secure Channel (SChannel).

Tabla 19 en la página 70 lista las propiedades que se deben definir en el objeto ConnectionFactory.

Tabla 19. Propiedades de ConnectionFactory para establecer conexiones seguras con un motor de mensajería del bus de integración de servicios WebSphere	
Nombre de la propiedad	**Descripción**
XMSC_WPM_SSL_CIPHER_SUITE	Nombre de la suite de cifrado que se debe utilizar en una conexión SSL o TLS con un motor de mensajería de WebSphere Servicio Integration Bus. El protocolo utilizado en la negociación de la conexión segura depende de la CipherSuite especificada. **Nota:** Esta propiedad está soportada en una aplicación .NET.
XMSC_WPM_SSL_KEYRING_LABEL	El certificado que se va a utilizar al autenticarse con el servidor. **Nota:** Esta propiedad está soportada en una aplicación .NET.

Lo siguiente es un ejemplo de propiedades de ConnectionFactory que se utilizan para establecer conexiones seguras con un motor de mensajería del bus de integración de servicios WebSphere:

```
cf.setStringProperty(XMSC_WPM_PROVIDER_ENDPOINTS, host_name:port_number:chain_name);
cf.setStringProperty(XMSC_WPM_SSL_KEY_REPOSITORY, key_repository_pathname);
cf.setStringProperty(XMSC_WPM_TARGET_TRANSPORT_CHAIN, transport_chain);
cf.setStringProperty(XMSC_WPM_SSL_CIPHER_SUITE, cipher_suite);
cf.setStringProperty(XMSC_WPM_SSL_KEYRING_STASH_FILE, stash_file_pathname);
```

Donde nombre_cadena se debería establecer en BootstrapTunneledSecureMessaging o BootstrapSecureMessaging, y número_puerto es el número del puerto donde el servidor de programa de arranque escucha solicitudes entrantes.

Lo siguiente es un ejemplo de propiedades de ConnectionFactory que se utilizan para establecer conexiones seguras con un motor de mensajería del bus de integración de servicios WebSphere, donde se han insertado valores de muestra:

```
  /* CF properties needed for an SSL connection */
    cf.setStringProperty(XMSC_WPM_PROVIDER_ENDPOINTS,"localhost:7286:BootstrapSecureMessaging");
    cf.setStringProperty(XMSC_WPM_TARGET_TRANSPORT_CHAIN,"InboundSecureMessaging");
    cf.setStringProperty(XMSC_WPM_SSL_KEY_REPOSITORY,"C:\\Program Files\\IBM\\gsk7\\bin\
\XMSkey.kdb");
    cf.setStringProperty(XMSC_WPM_SSL_KEYRING_STASH_FILE,"C:\\Program Files\\IBM\\gsk7\\bin\
\XMSkey.sth");
    cf.setStringProperty(XMSC_WPM_SSL_CIPHER_SUITE,"SSL_RSA_EXPORT_WITH_RC4_40_MD5");
```

Referencia relacionada

IConnectionFactory (para la interfaz .NET)
Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Propiedades de ConnectionFactory
Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Correlaciones de nombres CipherSuite y CipherSpec para conexiones a un WebSphere Servicio Integration Bus

Puesto que GSKit utiliza CipherSpecs en lugar de CipherSuites, los nombres de CipherSuite de estilo JSSE especificados en la propiedad XMSC_WPM_SSL_CIPHER_SUITE se deben correlacionar con los nombres de CipherSpec de estilo GSKit.

Tabla 20 en la página 71 lista la CipherSpec equivalente para cada CipherSuite reconocida.

Tabla 20. CipherSuites disponibles y sus CipherSpecs equivalentes

CipherSuite	**CipherSpec equivalente**
SSL_RSA_WITH_NULL_MD5	NULL_MD5
SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5	RC2_MD5_EXPORT
SSL_RSA_EXPORT_WITH_RC4_40_MD5	RC4_MD5_EXPORT
SSL_RSA_WITH_RC4_128_MD5	RC4_MD5_US
SSL_RSA_WITH_NULL_SHA	NULL_SHA
SSL_RSA_EXPORT1024_WITH_RC4_56_SHA	RC4_56_SHA_EXPORT1024
SSL_RSA_WITH_RC4_128_SHA	RC4_SHA_US
SSL_RSA_WITH_DES_CBC_SHA	DES_SHA_EXPORT
SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA	DES_SHA_EXPORT1024
SSL_RSA_FIPS_WITH_DES_CBC_SHA	FIPS_WITH_DES_CBC_SHA
SSL_RSA_WITH_3DES_EDE_CBC_SHA	TRIPLE_DES_SHA_US
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA	FIPS_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA	TLS_RSA_WITH_DES_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA	TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA	TLS_RSA_WITH_AES_256_CBC_SHA

Nota:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA está en desuso. Sin embargo, se sigue pudiendo utilizar para transferir hasta 32 GB de datos antes de que se termine la conexión con el error AMQ9288. Para evitar este error, tendrá que evitar utilizar el triple DES, o habilitar el restablecimiento de clave secreta cuando se utiliza esta CipherSpec.

Mensajes de XMS

Este secciónCapítulo describe la estructura y el contenido de los mensajes de XMS y explica cómo las aplicaciones procesan los mensajes de XMS.

Este secciónCapítulo contiene los temassecciones siguientes:

Referencia relacionada

IMessage (para la interfaz .NET)
Un objeto Message representa un mensaje que una aplicación envía o recibe. IMessage es una superclase para las clases de mensaje como, por ejemplo, IMapMessage.

Componentes del mensaje Un XMS

Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Cabecera
La cabecera de un mensaje contiene campos y todos los mensajes contienen el mismo conjunto de campos de cabecera. XMS y las aplicaciones utilizan los valores de los campos de cabecera para identificar y direccionar mensajes. Si desea más información sobre campos de cabecera. consulte "Campos de cabecera en mensajes de Un XMS" en la página 73.

Conjunto de propiedades
Las propiedades de un mensaje especifican información adicional sobre el mensaje. Aunque todos los mensajes tienen el mismo conjunto de campos de cabecera, cada mensaje puede tener un conjunto de propiedades diferente. Si desea más información, consulte "Propiedades de mensaje de Un XMS" en la página 74.

Body (Cuerpo)
El cuerpo de un mensaje contiene datos de aplicación. Si desea más información, consulte "Cuerpo de un mensaje de Un XMS" en la página 77.

Una aplicación puede seleccionar qué mensajes desea recibir. Mediante el uso de selectores de mensajes, que especifican los criterios de selección. Los criterios se pueden basar en los valores de determinados campos de cabecera y los valores de cualquiera de las propiedades de un mensaje. Si desea más información sobre selectores de mensajes, consulte "Selectores de mensaje" en la página 83.

Referencia relacionada

Campos de cabecera en mensajes de Un XMS
Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Propiedades de mensaje de Un XMS
XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Cuerpo de un mensaje de Un XMS

El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

Selectores de mensaje
La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ
Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Campos de cabecera en mensajes de Un XMS

Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Los nombres de estos campos de cabecera empiezan con el prefijo JMS. Si desea una descripción de los campos de cabecera del mensaje JMS, consulte *Java Message Service Specification, Versión 1.1*.

XMS implementa los campos de cabecera de mensaje JMS como atributos de un objeto Message. Cada campo de cabecera tiene sus propios métodos para establecer y obtener su valor. Si desea una descripción de estos métodos, consulte "IMessage" en la página 127. Un campo de cabecera siempre se puede leer y grabar.

La Tabla 21 en la página 73 lista los campos de cabecera de mensaje JMS e indica cómo se establece el valor de cada campo para un mensaje transmitido. Algunos de los campos se establecen automáticamente mediante XMS cuando una aplicación envía un mensaje o, en el caso de JMSRedelivered, cuando una aplicación recibe un mensaje.

Tabla 21. Campos de cabecera de mensaje JMS

Nombre del campo de cabecera de mensaje JMS	**Cómo se establece el valor para un mensaje transmitido (con el formato de *método* [*clase*])**
JMSCorrelationID	Establecer JMSCorrelationID [Message]
JMSDeliveryMode	Enviar [MessageProducer]
JMSDestination	Enviar [MessageProducer]
JMSExpiration	Enviar [MessageProducer]
JMSMessageID	Enviar [MessageProducer]
JMSPriority	Enviar [MessageProducer]
JMSRedelivered	Recibir [MessageConsumer]
JMSReplyTo	Establecer JMSReplyTo [Message]
JMSTimestamp	Enviar [MessageProducer]
JMSType	Establecer JMSType [Message]

Referencia relacionada

Componentes del mensaje Un XMS
Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Propiedades de mensaje de Un XMS
XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Cuerpo de un mensaje de Un XMS
El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

Selectores de mensaje
La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ
Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Propiedades de mensaje de Un XMS

XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Una aplicación XMS puede intercambiar mensajes con una aplicación WebSphere JMS porque XMS admite las propiedades predefinidas siguientes de un objeto Message:

- Las mismas propiedades definidas por JMS que admite WebSphere JMS. Los nombres de estas propiedades empiezan con el prefijo JMSX.
- Las mismas propiedades definidas por IBM que admite WebSphere JMS. Los nombres de estas propiedades empiezan con el prefijo JMS_IBM_.

Cada propiedad predefinida tiene dos nombres:

- Un nombre JMS, para una propiedad definida por JMS, o un nombre WebSphere JMS , para una propiedad definida por IBM.

  Esto es el nombre por el cual se conoce la propiedad en JMS o WebSphere JMS, y es también el nombre que se transmite con un mensaje que tenga esta propiedad. La aplicación Un XMS utiliza este nombre para identificar la propiedad en una expresión de selector de mensaje.
- Nombre Un XMS para identificar la propiedad en todas las situaciones, excepto en una expresión de selector de mensaje. Cada nombre XMS se define como una constante con nombre en la clase `IBM.XMS.XMSC`. El valor de la constante con nombre es el nombre JMS o WebSphere JMS correspondiente.

Además de las propiedades predefinidas, la aplicación Un XMS puede crear y utilizar su propio conjunto de propiedades de mensaje. Estas propiedades se denominan *propiedades definidas por aplicación.*

Después de que una aplicación haya creado un mensaje, las propiedades del mensaje se pueden leer y grabar. Las propiedades se siguen pudiendo leer y grabar después de que la aplicación envíe el mensaje. Cuando una aplicación recibe un mensaje, las propiedades del mensaje son de solo lectura. Si una aplicación llama al método `Clear Properties` de la clase Message cuando las propiedades de un mensaje son de sólo lectura, las propiedades pasan a ser legibles y grabables. El método también borra las propiedades.

El mensaje recibido, cuando se envía después de borrar las propiedades del mensaje, se comportará de una forma coherente con el comportamiento del envío de un WMQ XMS for .NET BytesMessage estándar con las propiedades de mensaje borradas.

Sin embargo, esto no se recomienda porque se perderán las propiedades siguientes:

- Valor de propiedad JMS_IBM_Encoding, que implica que los datos de mensaje no se pueden descodificar con sentido.
- Valor de propiedad JMS_IBM_Format, que implica que se rompería el encadenamiento de cabecera entre la cabecera de mensaje (MQMD o la MQRFH2 nueva) y las cabeceras existentes.

Referencia relacionada

Componentes del mensaje Un XMS
Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Campos de cabecera en mensajes de Un XMS
Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Cuerpo de un mensaje de Un XMS
El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

Selectores de mensaje

La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ
Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Propiedades definidas por JMS de un mensaje

XMS y WebSphere JMS admiten ambos varias propiedades definidas por JMS de un mensaje.

Tabla 22 en la página 75 lista las propiedades definidas por JMS de un mensaje que admiten XMS y, también, WebSphere JMS. Si desea una descripción de las propiedades definidas por JMS, consulte *Java Message Service Specification, Versión 1.1*. Las propiedades definidas por JMS no son válidas para una conexión en tiempo real con un intermediario.

La tabla especifica el tipo de datos de cada propiedad e indica cómo se establece el valor de la propiedad para un mensaje transmitido. Algunas de las propiedades se establecen automáticamente mediante XMS cuando una aplicación envía un mensaje o, en el caso de JMSXDeliveryCount, cuando una aplicación recibe un mensaje.

Tabla 22. Propiedades definidas por JMS de un mensaje

Nombre XMS de la propiedad definida de JMS	**Nombre JMS**	**Tipo de datos**	**Cómo se establece el valor para un mensaje transmitido (con el formato *método* [*clase*])**
JMSX_APPID	JMSXAppID	System.String	Enviar [MessageProducer]
JMSX_DELIVERY_COUNT	JMSXDeliveryCount	System.Int32	Recibir [MessageConsumer]
JMSX_GROUPID	JMSXGroupID	System.String	Establecer propiedad de serie [PropertyContext]
JMSX_GROUPSEQ	JMSXGroupSeq	System.Int32	Establecer propiedad de entero [PropertyContext]
JMSX_USERID	JMSXUserID	System.String	Enviar [MessageProducer]

Propiedades definidas por IBM de un mensaje

XMS y WebSphere JMS dan soporte a varias propiedades definidas por IBM de un mensaje.

Tabla 23 en la página 75 lista las propiedades definidas de IBM de un mensaje que están soportadas por XMS y WebSphere JMS. Para obtener más información sobre las propiedades definidas por IBM, consulte el manual *IBM WebSphere MQ - Utilización de Java* o la documentación de producto de WebSphere Application Server.

La tabla especifica el tipo de datos de cada propiedad e indica cómo se establece el valor de la propiedad para un mensaje transmitido. XMS establece automáticamente algunas de las propiedades cuando una aplicación envía un mensaje.

Tabla 23. Propiedades definidas por IBM de un mensaje

Nombre XMS de la propiedad definida de IBM	**Nombre WebSphere JMS**	**Tipo de datos**	**Cómo se establece el valor para un mensaje transmitido (con el formato *método* [*clase*])**
JMS_IBM_CHARACTER_SET	JMS_IBM_Character_Set	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_ENCODING	JMS_IBM_Encoding	System.Int32	Establecer propiedad de entero [PropertyContext]

Tabla 23. Propiedades definidas por IBM de un mensaje (continuación)

Nombre XMS de la propiedad definida de IBM	**Nombre WebSphere JMS**	**Tipo de datos**	**Cómo se establece el valor para un mensaje transmitido (con el formato *método* [*clase*])**
JMS_IBM_EXCEPTIONMESSAGE	JMS_IBM_ExceptionMessage	System.String	Recibir [MessageConsumer]
JMS_IBM_EXCEPTIONREASON	JMS_IBM_ExceptionReason	System.Int32	Recibir [MessageConsumer]
JMS_IBM_EXCEPTIONTIMESTAMP	JMS_IBM_ExceptionTimestamp	System.Int64	Recibir [MessageConsumer]
JMS_IBM_EXCEPTIONPROBLEM DESTINO	JMS_IBM_ExceptionProblemDestination	System.String	Recibir [MessageConsumer]
JMS_IBM_FEEDBACK	JMS_IBM_Feedback	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_FORMAT	JMS_IBM_Format	System.String	Establecer propiedad de serie [PropertyContext]
JMS_IBM_LAST_MSG_IN_GROUP	JMS_IBM_Last_Msg_In_Group	System.Boolean	Establecer propiedad de entero [PropertyContext]
JMS_IBM_MSGTYPE	JMS_IBM_MsgType	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_PUTAPPLTYPE	JMS_IBM_PutApplType	System.Int32	Enviar [MessageProducer]
JMS_IBM_PUTDATE	JMS_IBM_PutDate	System.String	Enviar [MessageProducer]
JMS_IBM_PUTTIME	JMS_IBM_PutTime	System.String	Enviar [MessageProducer]
JMS_IBM_REPORT_COA	JMS_IBM_Report_COA	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_COD	JMS_IBM_Report_COD	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_DISCARD_MSG	JMS_IBM_Report_Discard_Msg	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_EXCEPTION	JMS_IBM_Report_Exception	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_EXPIRATION	JMS_IBM_Report_Expiration	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_NAN	JMS_IBM_Report_NAN	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_PAN	JMS_IBM_Report_PAN	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_REPORT_PASS_CORREL_ ID	JMS_IBM_Report_Pass_Correl_ID	System.Int32	Establecer propiedad de entero [PropertyContext]

Tabla 23. Propiedades definidas por IBM de un mensaje (continuación)			
Nombre XMS de la propiedad definida de IBM	**Nombre WebSphere JMS**	**Tipo de datos**	**Cómo se establece el valor para un mensaje transmitido (con el formato *método* [*clase*])**
JMS_IBM_REPORT_PASS_MSG_ID	JMS_IBM_Report_Pass_Msg_ID	System.Int32	Establecer propiedad de entero [PropertyContext]
JMS_IBM_SYSTEM_MESSAGEID	JMS_IBM_System_MessageID	System.String	Enviar [MessageProducer]

Propiedades de un mensaje definidas por la aplicación

Una aplicación XMS puede crear y utilizar su propio conjunto de propiedades de mensaje. Cuando una aplicación envía un mensaje, estas propiedades también se transmiten con el mensaje. Una aplicación receptora, que utiliza selectores de mensajes, puede seleccionar qué mensajes desea recibir basándose en los valores de estas propiedades.

Para permitir que una aplicación WebSphere JMS seleccione y procese mensajes enviados por una aplicación XMS, el nombre de una propiedad definida por la aplicación debe cumplir las reglas para crear identificadores en expresiones de selector de mensaje, tal como se describe en el manual *IBM WebSphere MQ Utilización de Java.* El valor de una propiedad definida por la aplicación debe tener uno de los tipos de datos siguientes: System.Boolean, System.SByte, System.Int16, System.Int32, System.Int64, System.Float, System.Double o System.String.

Cuerpo de un mensaje de Un XMS

El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

XMS admite cinco tipos de cuerpo de mensaje:

Bytes
El cuerpo contiene una corriente de bytes. Un mensaje con este tipo de cuerpo se llama un *mensaje de bytes* La interfaz IBytesMessage contiene los métodos para procesar el cuerpo de un mensaje de bytes. Si desea más información, consulte “Mensajes de bytes” en la página 79.

Correlación
El cuerpo contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado. Un mensaje con este tipo de cuerpo se llama un *mensaje de correlación*. La interfaz IMapMessage contiene los métodos para procesar el cuerpo de un mensaje de correlación. Si desea más información, consulte “Mensajes de correlación” en la página 80.

Objeto
El cuerpo contiene un objeto serializado Java o .NET. Un mensaje con este tipo de cuerpo se llama un *mensaje de objeto* La interfazIObjectMessage contiene los métodos para procesar el cuerpo de un mensaje de objeto. Si desea más información, consulte “Mensajes de objeto” en la página 81.

Corriente de datos (Stream)
El cuerpo contiene una corriente de valores, donde cada valor tiene un tipo de datos asociado. Un mensaje con este tipo de cuerpo se llama un *mensaje de corriente de datos*. La interfaz IStreamMessage contiene los métodos para procesar el cuerpo de un mensaje de corriente de datos. Si desea más información, consulte “Mensajes de corriente de datos” en la página 81.

Texto
El cuerpo contiene una serie. Un mensaje con este tipo de cuerpo se llama un *mensaje de texto* La interfaz ITextMessage contiene los métodos para procesar el cuerpo de un mensaje de texto. Si desea más información, consulte “Mensajes de texto” en la página 82.

La interfaz IMessage es el padre de todos los objetos de mensaje y se puede utilizar en funciones de mensajería para representar cualquiera de los tipos de mensaje de XMS.

Si desea más información sobre el tamaño y los valores máximo y mínimo de cada uno de estos tipos de datos, consulte Tabla 5 en la página 42.

Si desea más información sobre los tipos de datos necesarios para elementos de datos de aplicación escritos en el cuerpo de un mensaje y sobre los cinco tipos de mensaje de cuerpo, consulte los subtemas.

Referencia relacionada

Componentes del mensaje Un XMS
Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Campos de cabecera en mensajes de Un XMS
Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Propiedades de mensaje de Un XMS
XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Selectores de mensaje
La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ
Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Tipos de datos para elementos de datos de aplicación

Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Por este motivo, cada dato de aplicación escrito en el cuerpo de un mensaje por la aplicación Un XMS debe tener uno de los tipos de datos listados en la Tabla 24 en la página 78. Para cada tipo de datos, la tabla muestra el tipo de datos Java compatible. XMS proporciona los métodos para escribir elementos de datos de aplicación solo con estos tipos de datos.

Tabla 24. Tipos de datos XMS que son compatibles con tipos de datos Java

Tipo de datos XMS	Representa	Tipo de datos Java compatible
System.Boolean	El valor booleano true o false	boolean
System.Char16	Carácter de doble byte	carácter
System.SByte	Entero de 8 bits firmado	byte
System.Int16	Entero de 16-bits firmado	corto
System.Int32	Entero de 32-bits firmado	int
System.Int64	Entero de 64-bits firmado	largo
System.Float	Número de coma flotante firmado	flotante
System.Double	Número de coma flotante de precisión doble firmado	doble
System.String	Serie de caracteres	Cadena

Si desea más información sobre el tamaño, el valor máximo y el valor mínimo de cada uno de estos tipos de datos, consulte “Tipos primitivos de XMS” en la página 42.

Conceptos relacionados

Atributos y propiedades de objetos
Un objeto XMS puede tener atributos y propiedades, que son característicos del objeto, que se implementan de formas diferentes.

Tipos primitivos de XMS
XMS proporciona equivalentes de los ochos tipos primitivos de Java (byte, short, int, long, float, double, char y boolean). Esto permite el intercambio de mensajes entre XMS y JMS sin pérdida ni corrupción de datos.

Conversión implícita de un valor de propiedad de un tipo de datos a otro
Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Referencia relacionada

Mensajes de bytes
El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Mensajes de correlación
El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de objeto
El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

Mensajes de bytes

El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Los mensajes de bytes son útiles si la aplicación de Un XMS necesita intercambiar mensajes con aplicaciones que no utilizan XMS o la interfaz de programación de aplicaciones de JMS.

Después de que una aplicación cree un mensaje debytes, el cuerpo del mensaje es de solo escritura. La aplicación ensambla los datos de aplicación en el cuerpo llamando a los métodos de escritura apropiados de la interfaz IBytesMessage para .NET. Cada vez que la aplicación escribe un valor en la corriente de mensajes de bytes, el valor se ensambla inmediatamente después del valor anterior escrito por la aplicación. XMS mantiene un cursor interno para recordar la posición del último byte que se ensambló.

Cuando la aplicación envía el mensaje, el cuerpo del mensaje se convierte en de solo lectura. De esta forma, la aplicación puede enviar el mensaje de forma repetida.

Cuando una aplicación recibe un mensaje de bytes, el cuerpo del mensaje es de solo lectura. La aplicación puede utilizar los métodos de lectura apropiados de la interfaz IBytesMessage para leer el contenido de la corriente de mensajes de bytes. La aplicación lee los bytes de la secuencia, y XMS mantiene un cursor interno para recordar la posición del último byte que se leyó.

Si una aplicación llama al método Restablecer de la interfaz IBytesMessage cuando el cuerpo de un mensaje de bytes se puede grabar, el cuerpo pasa a ser de solo lectura. El método también vuelve a colocar el cursor al inicio de la corriente de mensajes de bytes.

Si una aplicación llama al método `Clear Body` de la interfaz IMessage para .NET cuando el cuerpo de un mensaje de bytes es de sólo lectura, el cuerpo pasa a ser grabable. El método también borra el cuerpo.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Mensajes de correlación

El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de objeto
El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

IBytesMessage (para la interfaz .NET)
Un mensaje de bytes es un mensaje cuyo cuerpo está formado por una corriente de bytes.

Mensajes de correlación

El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

En cada par de nombre-valor, el nombre es una serie que identifica el valor, y el valor es un elemento de datos de aplicación que tiene uno de los tipos de datos de XMS listados en Tabla 24 en la página 78. El orden de los partes de nombre-valor no está definido. La clase MapMessage contiene los métodos para establecer y obtener pares de nombre-valor.

Una aplicación puede acceder a un par de nombre-valor de forma aleatoria especificando su nombre.

Una aplicación .NET puede utilizar la propiedad MapNames para obtener una enumeración de los nombres en el cuerpo del mensaje de correlación.

Cuando una aplicación obtiene el valor de un par de nombre-valor, el valor se puede convertir mediante XMS en otro tipo de datos. Por ejemplo, para obtener un entero del cuerpo de un mensaje de correlación, una aplicación puede llamar al método GetString de la clase MapMessage, que devuelve el entero como una serie. Las conversiones soportadas son las mismas que las que están soportadas cuando XMS convierte un valor de propiedad de un tipo a otro. Si desea más información sobre las conversiones soportadas, consulte “Conversión implícita de un valor de propiedad de un tipo de datos a otro” en la página 42.

Después de que una aplicación crea un mensaje de correlación, el cuerpo del mensaje se puede leer y grabar. El cuerpo se sigue pudiendo leer y escribir después de que la aplicación envíe el mensaje. Cuando una aplicación recibe un mensaje de correlación, el cuerpo del mensaje es de solo lectura. Si una aplicación llama al método Borrar cuerpo de la clase Message cuando el cuerpo de un mensaje de correlación es de solo lectura, el cuerpo pasa poderse leer y grabar. El método también borra el cuerpo.

Conceptos relacionados

Conversión implícita de un valor de propiedad de un tipo de datos a otro
Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Mensajes de bytes
El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Mensajes de objeto
El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Mensajes de corriente de datos

El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

IMapMessage (para la interfaz .NET)
Un mensaje de correlación es un mensaje cuyo cuerpo está formado por un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de objeto

El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Una aplicación Un XMS puede recibir un mensaje de objeto, cambiar sus campos de cabecera y propiedades, y luego enviarlo a otro destino. Una aplicación también puede copiar el cuerpo de un mensaje de objeto y utilizarlo para formar otro mensaje de objeto. XMS trata el cuerpo de un mensaje de objeto como una matriz de bytes.

Después de que una aplicación crea un mensaje de objeto, el cuerpo del mensaje se puede leer y escribir. El cuerpo se sigue pudiendo leer y escribir después de que la aplicación envíe el mensaje. Cuando una aplicación recibe un mensaje de objeto, el cuerpo del mensaje es de solo lectura. Si una aplicación llama al método `Clear Body` de la interfaz IMessage para .NET cuando el cuerpo de un mensaje de objeto es de sólo lectura, el cuerpo pasa a ser legible y grabable. El método también borra el cuerpo.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Mensajes de bytes
El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Mensajes de correlación
El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

IObjectMessage (para la interfaz .NET)
Un mensaje de objeto es un mensaje cuyo cuerpo consta de un objeto Java o .NET serializado.

Mensajes de corriente de datos

El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

El tipo de datos de un valor es uno de los tipos de datos XMS listados en Tabla 24 en la página 78.

Después de que una aplicación haya creado un mensaje de corriente de datos, se puede escribir en el cuerpo del mensaje. La aplicación ensambla los datos de aplicación en el cuerpo llamando a los métodos de escritura apropiados de la interfaz IStreamMessage para .NET. Cada vez que la aplicación escribe un valor en la corriente de datos del mensaje, el valor y su tipo de datos se ensamblan inmediatamente después del valor anterior escrito por la aplicación. XMS mantiene un cursor interno para recordar la posición del último valor que se ha ensamblado.

Cuando la aplicación envía el mensaje, el cuerpo del mensaje se convierte en de solo lectura. De esta forma, la aplicación puede enviar el mensaje varias veces.

Cuando una aplicación recibe un mensaje de corriente de datos, el cuerpo del mensaje es de solo lectura. La aplicación puede utilizar los métodos de lectura apropiados de la interfaz IStreamMessage para .NET para leer el contenido de la corriente de datos de mensaje. La aplicación lee los valores en secuencia, y XMS mantiene un cursor interno para recordar la posición del último valor que leído.

Cuando una aplicación lee un valor de la corriente de datos de mensaje, el valor se puede convertir mediante XMS a otro tipo de datos. Por ejemplo, para leer un entero de la corriente de datos de mensaje, una aplicación puede llamar al método ReadString, que devuelve el entero como una serie. Las conversiones soportadas son las mismas que las que están soportadas cuando XMS convierte un valor de propiedad de un tipo a otro. Si desea más información sobre las conversiones soportadas, consulte "Conversión implícita de un valor de propiedad de un tipo de datos a otro" en la página 42.

Si se produce un error mientras una aplicación está intentando leer un valor de la corriente de datos de mensaje, el cursor no avanza. La aplicación puede recuperarse del error intentando leer el valor como otro tipo de datos.

Si una aplicación llama al método Restablecer de la interfaz IStreamMessage para .NET cuando el cuerpo de un mensaje de corriente de datos es de solo escritura, el cuerpo pasa a ser de solo lectura. El método también vuelve a colocar el cursor al inicio de la corriente de datos de mensaje.

Si una aplicación llama al método `Clear Body` de la interfaz IMessage para .NET cuando el cuerpo de un mensaje de corriente es de sólo lectura, el cuerpo pasa a ser de sólo escritura. El método también borra el cuerpo.

Conceptos relacionados

Conversión implícita de un valor de propiedad de un tipo de datos a otro
Cuando una aplicación obtiene el valor de una propiedad, el valor se puede convertir mediante XMS a otro tipo de datos. Muchas normas rigen qué conversiones están soportadas y cómo XMS realiza las conversiones.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Mensajes de bytes
El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Mensajes de correlación
El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de objeto
El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

IStreamMessage (para la interfaz .NET)
Un mensaje de corriente de datos es un mensaje cuyo cuerpo está formado por una corriente de valores, donde cada valor tiene un tipo de datos asociado. El contenido del cuerpo se escribe y se lee de forma secuencial.

Mensajes de texto

El cuerpo de un mensaje de texto contiene una serie.

Después de que una aplicación crea un mensaje de texto, el cuerpo del mensaje se puede leer y escribir. El cuerpo se sigue pudiendo leer y escribir después de que la aplicación envíe el mensaje. Cuando una aplicación recibe un mensaje de texto, el cuerpo del mensaje es de solo lectura. Si una aplicación llama al método Borrar cuerpo de la interfaz IMessage para .NET cuando el cuerpo de un mensaje de texto es de solo lectura, el cuerpo pasa a poderse leer y escribir. El método también borra el cuerpo.

Referencia relacionada

Tipos de datos para elementos de datos de aplicación
Para garantizar que una aplicación XMS puede intercambiar mensajes con una aplicación IBM WebSphere MQ classes for JMS, ambas aplicaciones deben poder interpretar los datos de aplicación del cuerpo de un mensaje de la misma forma.

Mensajes de bytes
El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Mensajes de correlación
El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Mensajes de objeto
El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

ITextMessage (para la interfaz .NET)
Un mensaje de texto es un mensaje cuyo cuerpo está formado por una serie.

Selectores de mensaje

La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Cuando una aplicación crea un consumidor de mensajes, puede asociar una expresión de selector de mensajes al consumidor. La expresión de selector de mensajes especifica los criterios de selección.

Cuando una aplicación se conecta a un gestor de colas de IBM WebSphere MQ V7.0, la selección de mensajes se realiza en el extremo del gestor de colas. XMS no realiza ninguna selección y, simplemente, entrega el mensaje que ha recibido del gestor de colas proporcionando, de esta forma, un mejor rendimiento.

Sin embargo, cuando una aplicación se conecta a IBM WebSphere MQ V6.0 e inferior, WebSphere Intermediario de sucesoso WebSphere Message Broker, WebSphere Service Integration Bus XMS determina si cada mensaje de entrada cumple los criterios de selección. Si un mensaje cumple los criterios de selección, XMS entrega el mensaje al consumidor de mensajes. Si un mensaje no cumple los criterios de selección, XMS no entrega el mensaje y, en el dominio punto a punto, el mensaje permanece en la cola.

Una aplicación puede crear más de un consumidor de mensajes, cada uno con su propia expresión de selector de mensajes. Si un mensaje entrante cumple los criterios de selección de más de un consumidor de mensajes, XMS entrega el mensaje a cada uno de estos consumidores.

Una expresión de selector de mensajes puede hacer referencia a las propiedades de mensaje siguientes:

- Propiedades definidas por JMS
- Propiedades definidas por IBM
- Propiedades definidas por aplicación

También puede hacer referencia a los campos de cabecera de mensaje siguientes:

- JMSCorrelationID
- JMSDeliveryMode
- JMSMessageID
- JMSPriority
- JMSTimestamp
- JMSType

Sin embargo, una expresión de selector de mensajes no puede hacer referencia a los datos del cuerpo de un mensaje.

A continuación, se muestra un ejemplo de expresión de selector de mensajes:

```
JMSPriority > 3 AND manufacturer = 'Jaguar' AND model in ('xj6','xj12')
```

XMS entrega un mensaje a un consumidor de mensajes con esta expresión de selector de mensajes solo si el mensaje tiene una prioridad mayor que 3; una propiedad definida por la aplicación, manufacturer (fabricante), con un valor `Jaguar;` y otra propiedad definida por la aplicación, model (modelo), con un valor de `xj6` o `xj12.`

Las reglas de sintaxis para formar una expresión de selector de mensajes en XMS son las mismas que las de IBM WebSphere MQ classes for JMS. Para obtener información sobre cómo crear una expresión de selector de mensajes, consulte el manual *WebSphere MQ - Utilización de Java.* Observe que, en una expresión de selector de mensajes, los nombres de las propiedades definidas por JMS deben ser los nombres de JMS, y los nombres de las propiedades definidas por IBM deben ser los nombres de IBM WebSphere MQ classes for JMS. No puede utilizar los nombres XMS en una expresión de selector de mensajes.

Referencia relacionada

Componentes del mensaje Un XMS
Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Campos de cabecera en mensajes de Un XMS
Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Propiedades de mensaje de Un XMS
XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Cuerpo de un mensaje de Un XMS
El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ
Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Correlación de mensajes de XMS con mensajes de IBM WebSphere MQ

Los campos de cabecera JMS y propiedades de un mensaje Un XMS se correlacionan con campos de las estructuras de cabecera de un mensaje IBM WebSphere MQ.

Cuando una aplicación Un XMS se conecta a un gestor de colas IBM WebSphere MQ, los mensajes enviados al gestor de colas se correlacionan con mensajes IBM WebSphere MQ de la misma forma que los mensajes IBM WebSphere MQ classes for JMS se correlacionan con mensajes IBM WebSphere MQ en casos similares.

Si la propiedad XMSC_WMQ_TARGET_CLIENT de un objeto Destination está establecida en XMSC_WMQ_TARGET_DEST_JMS, los campos de cabecera JMS y las propiedades de un mensaje enviado al destino se correlacionan en campos de las estructuras de cabecera MQMD y MQRFH2 del mensaje IBM WebSphere MQ. Establecer la propiedad XMSC_WMQ_TARGET_CLIENT de esta forma da por supuesto que la aplicación que recibe el mensaje puede manejar una cabecera MQRFH2. Por lo tanto, la aplicación receptora podría ser otra aplicación XMS, una aplicación IBM WebSphere MQ classes for JMS o una aplicación IBM WebSphere MQ nativa que se ha diseñado para manejar una cabecera MQRFH2.

Si la propiedad XMSC_WMQ_TARGET_CLIENT de un objeto Destination está establecida en XMSC_WMQ_TARGET_DEST_MQ en su lugar, los campos de cabecera JMS y las propiedades de un mensaje enviado al destino se correlacionan en los campos de la estructura de cabecera MQMD del mensaje IBM WebSphere MQ. El mensaje no contiene una cabecera MQRFH2, y se ignora cualquier campo de cabecera JMS y propiedad que no se pueda correlacionar en los campos de la estructura

de cabecera MQMD. Por lo tanto, la aplicación que recibe el mensaje puede ser una aplicación IBM WebSphere MQ nativa que no se ha diseñado para manejar una cabecera MQRFH2.

Los mensajes IBM WebSphere MQ recibidos de un gestor de colas se correlacionan en los mensajes XMS de la misma forma que los mensajes IBM WebSphere MQ se correlacionan en los mensajes IBM WebSphere MQ classes for JMS en circunstancias similares.

Si un mensaje IBM WebSphere MQ entrante tiene una cabecera MQRFH2, el mensaje XMS resultante tiene un cuerpo cuyo tipo se determina mediante el valor de la propiedad **Msd** incluida en la carpeta `mcd` de la cabecera MQRFH2. Si la propiedad **Msd** no está presente en la cabecera MQRFH2, o si el mensaje IBM WebSphere MQ no tiene ninguna cabecera MQRFH2, el mensaje XMS resultante tiene un cuerpo cuyo tipo se determina mediante el valor del campo *`Format`* en la cabecera MQMD. Si el campo *`Format`* está establecido en MQFMT_STRING, el mensaje XMS es un mensaje de texto. De lo contrario, el mensaje XMS es un mensaje de bytes. Si el mensaje IBM WebSphere MQ no tiene ninguna cabecera MQRFH2, solo se establecen los campos de cabecera JMS y las propiedades que se pueden derivar de campos de la cabecera MQMD.

Para obtener más información sobre la correlación de mensajes de IBM WebSphere MQ classes for JMS con mensajes de IBM WebSphere MQ , consulte *IBM WebSphere MQ Utilización de Java*.

Referencia relacionada

Componentes del mensaje Un XMS
Un mensaje Un XMS consta de una cabecera, un conjunto de propiedades y un cuerpo.

Campos de cabecera en mensajes de Un XMS
Para permitir que una aplicación Un XMS intercambie mensajes con una aplicación WebSphere JMS, la cabecera del mensaje Un XMS contiene los campos de cabecera del mensaje JMS.

Propiedades de mensaje de Un XMS
XMS es compatible con tres clases de propiedades de mensaje: propiedades definidas por JMS, propiedades definidas por IBM y propiedades definidas por la aplicación.

Cuerpo de un mensaje de Un XMS
El cuerpo de un mensaje contiene datos de aplicación. Sin embargo, un mensaje puede no tener cuerpo, y estar formado solo de los campos de cabecera y las propiedades.

Selectores de mensaje
La aplicación Un XMS utiliza selectores de mensajes para seleccionar los mensajes que desea recibir.

Lectura y escritura del descriptor de mensaje desde una aplicación Message Service Client for .NET

Puede acceder a todos los campos del descriptor de mensaje (MQMD) de un mensaje de IBM WebSphere MQ excepto StrucId y Version. Existe acceso de lectura para BackoutCount, pero no de escritura. Esta característica sólo está disponible cuando se conecta a un IBM WebSphere MQ gestor de colas Versión 6 y superior, y se controla mediante las propiedades de destino descritas más adelante.

Los atributos de mensaje proporcionados por Message Service Client for .NET facilitan aplicaciones XMS para establecer campos MQMD y, también, para dirigir aplicaciones IBM WebSphere MQ.

Existen algunas restricciones cuando se utiliza la mensajería de Publicar/Suscribir. Por ejemplo, campos MQMD como MsgID y CorrelId, si están establecidos, se ignoran.

La función descrita en este tema no está disponible para la mensajería de Publicar/Suscribir cuando se conecta a un gestor de colas de IBM WebSphere MQ V6. Tampoco está disponible cuando la propiedad **PROVIDERVERSION** está establecida en 6.

Acceso a datos de mensaje de IBM WebSphere MQ desde una aplicación de Message Service Client para .NET

Puede acceder a los datos completos de mensaje de IBM WebSphere MQ incluyendo la cabecera MQRFH2 (si está presente) y cualquier otra cabecera de IBM WebSphere MQ (si está presente) en una aplicación Message Service Client for .NET como cuerpo de un JMSBytesMessage.

La función descrita en este tema está disponible solamente si se conecta con un gestor de colas de IBM WebSphere MQ perteneciente a la Versión 7 o posterior y el proveedor de mensajería de IBM WebSphere MQ está en la modalidad normal.

Las propiedades del objeto Destination determinan la forma en que la aplicación de XMS accede a la totalidad de un mensaje de IBM WebSphere MQ (incluida la cabecera MQRFH2, si existe) como cuerpo de un JMSBytesMessage.

Resolución de problemas

Este secciónCapítulo proporciona información para ayudarle a detectar y tratar problemas al utilizar Message Service Client for .NET.

Este secciónCapítulo proporciona información para ayudarle en la determinación de problemas para aplicaciones XMS, y describe cómo configurar la captura de datos en primer error (FFDC) y el rastreo para aplicaciones .NET.

Este secciónCapítulo contiene los temassecciones siguientes:

- "Configuración del rastreo para aplicaciones .NET" en la página 86
- "Configuración de FFDC para aplicaciones .NET" en la página 90
- "Consejos para la resolución de problemas" en la página 91

Configuración del rastreo para aplicaciones .NET

Para aplicaciones XMS .NET, puede configurar el rastreo a partir de un archivo de configuración de aplicación, así como a partir de variables de entorno de XMS. Puede seleccionar los componentes que desea rastrear. Normalmente, el rastreo se utiliza bajo la supervisión del servicio de soporte de IBM.

El rastreo para XMS .NET se basa en la infraestructura de rastreo estándar de .NET.

Todo el rastreo, excepto el rastreo de errores, está inhabilitado de forma predeterminada. Puede activar el rastreo y configurar los valores de rastreo en cualquiera de las formas siguientes:

- Mediante el uso de un archivo de configuración de aplicación con un nombre que está formado por el nombre del programa ejecutable con el que está relacionado el archivo, con el sufijo `.config`. Por ejemplo, el archivo de configuración de aplicación para text.exe tendría el nombre text.exe.config. Utilizar un archivo de configuración de aplicación es la forma preferida para habilitar el rastreo para aplicaciones XMS .NET. Si desea más detalles, consulte "Configuración del rastreo mediante un archivo de configuración de aplicación" en la página 87.
- Mediante el uso de variables de entorno de XMS con respecto a aplicaciones C o C++ de XMS. Si desea más detalles, consulte "Configuración del rastreo utilizando variables de entorno XMS" en la página 89.

El archivo de rastreo activo tiene un nombre con el formato xms_trace <*PID*> .log donde <*PID*> representa el ID de proceso de la aplicación. El tamaño del archivo de rastreo activo está delimitado, de forma predeterminada, a 20 MB. Cuando se alcanza este límite, el archivo se renombra y se archiva. Las copias archivadas tienen nombres con el formato xms_trace <*PID*> *_YY.MM.DD_HH.MM.SS*.log

De forma predeterminada, el número de archivos de rastreo que se conservan es cuatro, es decir, un archivo activo y tres archivos archivados. Estos cuatro archivos se utilizan como un almacenamiento intermedio de registro hasta que se detiene la aplicación, con el archivo más antiguo eliminado y sustituido por el archivo más nuevo. Puede cambiar el número de archivos de rastreo especificando un número diferente en el archivo de configuración de aplicación. Sin embargo, debe haber al menos dos archivos (un archivo activo y un archivo archivado).

Están disponibles dos formatos de archivo de rastreo:

- Los archivos de rastreo de formato básico son legibles por el usuario, con un formato de WebSphere Application Server. Este formato es el formato de archivo de rastreo predeterminado. El formato básico no es compatible con las herramientas del analizador de rastreo.

- Los archivos de rastreo de formato avanzado son compatibles con las herramientas del analizador de rastreo. Debe especificar que desea generar archivos de rastreo en formato avanzado en el archivo de configuración de aplicación.

Las entradas de rastreo contienen la información siguiente:

- La fecha y hora cuando se registró el rastreo
- El nombre de clase
- El tipo de rastreo
- El mensaje de rastreo

El ejemplo siguiente muestra un exacto de algún rastreo:

```
[09/11/2005 14:33:46:914276]    00000004    IBM.XMS.Comms.IoRequest    >    Allocate    Entry
[09/11/2005 14:33:46:914276]    00000004    IBM.XMS.Comms.IoRequest    >    Initialize    Entry
[09/11/2005 14:33:46:914276]    00000004    IBM.XMS.Comms.IoRequest    <    Initialize    Exit
[09/11/2005 14:33:46:914276]    00000004    IBM.XMS.Comms.IoRequest    <    Allocate    Exit
```

En el ejemplo anterior, el formato es:

```
[Date Time:Microsecs]         Thread-id    Classname        Trace-type    Methodname    Entry
or Exit
```

Donde `Trace-type` es:

> para Entrada
< para Salida
d para Información de depuración

Configuración del rastreo mediante un archivo de configuración de aplicación

La forma preferida de configurar el rastreo para aplicaciones XMS .NET es con un archivo de configuración de aplicación. La sección de rastreo de este archivo incluye parámetros que definen lo que se va a rastrear, la ubicación del archivo de rastreo y el tamaño máximo permitido, el número de archivos de rastreo utilizado y el formato del archivo de rastreo.

Para activar el rastreo utilizando el archivo de configuración de aplicación, simplemente coloque el archivo en el mismo directorio que el archivo ejecutable para la aplicación.

El rastreo se puede habilitar por componente y por tipo de rastreo. También es posible activar el rastreo para todo un grupo de rastreo. Puede activar el rastreo para componentes de una jerarquía ya sea de forma individual o colectiva. Los tipos de rastreo disponibles incluyen:

- Rastreo de depuración
- Rastreo de excepción
- Avisos, mensajes informativos y mensajes de error
- Rastreo de entrada y salida de método

El ejemplo siguiente muestra los valores de rastreo definidos en la sección de rastreo de un archivo de configuración de aplicación:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <configSections>
        <sectionGroup name="IBM.XMS">
            <section name="Trace"
                     type="System.Configuration.SingleTagSectionHandler" />
        </sectionGroup>
    </configSections>

    <IBM.XMS>
        <Trace traceSpecification="*=all=enabled" traceFilePath=""
               traceFileSize="20000000" traceFileNumber="3"
               traceFormat="advanced" />
```

```
    </IBM.XMS>
  </configuration>
```

Tabla 25 en la página 88 describe los valores de parámetro de forma más detallada.

Tabla 25. Valores de parámetro de rastreo del archivo de configuración de aplicación

Parámetro	Descripción
traceSpecification=<ComponentName>=<type>=<state>	<ComponentName> es el nombre de la clase que desea rastrear. Puede utilizar un carácter comodín * en este nombre. Por ejemplo, *=all=enabled especifica que desea rastrear todas las clases y IBM.XMS.impl.*=all=enabled especifica que solo requiere el rastreo de API. <type> puede ser cualquiera de los siguientes tipos de rastreo: • Todo • depurar • suceso • EntryExit <state> puede estar habilitado o inhabilitado. Puede unir en una serie varios elementos de rastreo utilizando un delimitador ':' (dos puntos).
traceFilePath="<filename>"	Si no especifica un valor traceFilePath, o si traceFilePath está presente pero contiene una serie vacía, el archivo de rastreo se coloca en el directorio actual. Para almacenar el archivo de rastreo en un directorio especificado, especifique el nombre de directorio en el traceFilePath, por ejemplo: `traceFilePath="c:\somepath"`
traceFileSize="<size>"	El tamaño máximo permitido del archivo de rastreo. Cuando un archivo alcanza este tamaño, se archiva y se renombra. El máximo predeterminado es 20 KB, que se especifica como traceFileSize="20000000".
traceFileNumber="<number>"	El número de archivos de rastreo que se van a conservar. El valor predeterminado es 4 (un archivo activo y 3 archivos de archivado). El número mínimo permitido es 2.
traceFormat="<format>"	El formato de rastreo predeterminado es el básico. Los archivos de rastreo se generan en este formato si especifica traceFormat="basic", o si no especifica un traceFormat, o si el traceFormat está presente pero contiene una serie vacía. Si necesita un rastreo que sea compatible con las herramientas del analizador de rastreo, debe especificar traceFormat="advanced".

Los valores de rastreo del archivo de configuración de aplicación son dinámicos, y se vuelven a leer cada vez que el archivo se guarda o sustituye. Si se encuentran errores en el archivo una vez que se ha editado, los valores del archivo de rastreo se revierten a sus valores predeterminados.

Conceptos relacionados

Configuración del rastreo utilizando variables de entorno XMS
Como alternativa al uso de un archivo de configuración de aplicación, puede activar el rastreo utilizando variables de entorno XMS. Estas variables de entorno solo se utilizan si no hay ninguna especificación de rastreo en el archivo de configuración de aplicación.

Configuración del rastreo utilizando variables de entorno XMS

Como alternativa al uso de un archivo de configuración de aplicación, puede activar el rastreo utilizando variables de entorno XMS. Estas variables de entorno solo se utilizan si no hay ninguna especificación de rastreo en el archivo de configuración de aplicación.

Para configurar el rastreo para una aplicación XMS .NET, establezca las variables de entorno siguientes antes de ejecutar la aplicación.

Tabla 26. Valores de variable de entorno para el rastreo .NET

Variables de entorno	Valor predeterminado	Valores	Significado
XMS_TRACE_ON	No aplicable	No aplicable: se ignora el valor de esta variable	Si XMS_TRACE_ON está establecido, todo el rastreo está habilitado, de forma predeterminado.
XMS_TRACE_FILE_PATH	Directorio de trabajo actual	/dirpath/	La vía de acceso de directorio en la que se escriben registros FFDC y de rastreo. XMS crea archivos de rastreo y FFDC en el directorio de trabajo actual, a menos que especifique una ubicación alternativa. Puede especificar una ubicación alternativa estableciendo la variable de entorno XMS_TRACE_FILE_PATH en el nombre de vía de acceso completo del directorio, donde desea que XMS cree los archivos de rastreo y FFDC. Debe establecer la variable de entorno antes de iniciar la aplicación que desea rastrear. Debe asegurarse de que el identificador de usuario bajo el cual se ejecuta la aplicación tiene la autoridad para escribir en el directorio donde XMS crea los archivos de rastreo y FFDC.

Tabla 26. Valores de variable de entorno para el rastreo .NET (continuación)			
Variables de entorno	**Valor predeterminado**	**Valores**	**Significado**
XMS_TRACE_FORMAT	BÁSICO	BASIC, ADVANCED	Especifica el formato de rastreo necesario, que puede ser BASIC o ADVANCED. El formato predeterminado esBASIC. El formato ADVANCED es compatible con las herramientas de analizador de rastreo.
XMS_TRACE_SPECIFICATION	No aplicable	Consulte "Configuración del rastreo mediante un archivo de configuración de aplicación" en la página 87	Sustituye la especificación de rastreo, que sigue el formato especificado en "Configuración del rastreo mediante un archivo de configuración de aplicación" en la página 87.

Conceptos relacionados

Configuración del rastreo mediante un archivo de configuración de aplicación
La forma preferida de configurar el rastreo para aplicaciones XMS .NET es con un archivo de configuración de aplicación. La sección de rastreo de este archivo incluye parámetros que definen lo que se va a rastrear, la ubicación del archivo de rastreo y el tamaño máximo permitido, el número de archivos de rastreo utilizado y el formato del archivo de rastreo.

Configuración de FFDC para aplicaciones .NET

Para la implementación de .NET de XMS, se genera un archivo FFDC para cada FFDC.

Los archivos FFDC (First Failure Data Capture) se almacenan en archivos de texto legibles para los usuarios. Estos archivos tienen nombres con el formato `xmsffdc<`*`processID`*`>_<`*`Date`*`>T<`*`Timestamp`*`>.txt`. Un ejemplo de un nombre de archivo es `xmsffdc264_2006.01.06T13.18.52.990955.txt`. La indicación de fecha y hora contiene la resolución de microsegundos.

Los archivos empiezan con la fecha y hora cuando se ha producido la excepción, seguidas por el tipo de excepción. Los archivos incluyen un probeId corto exclusivo, que se pueden utilizar para localizar dónde se ha producido esta FFDC.

No es necesario que realice ninguna configuración para activar la FFDC. De forma predeterminada, todos los archivos deFFDC se escriben en el directorio actual. Sin embargo, si es necesario, puede especificar un directorio diferente cambiando ffdcDirectory en la sección Rastreo del archivo de configuración de aplicación. En el ejemplo siguiente, todos los archivos de rastreo se registran en el directorio c:client\ffdc:

```
<IBM.XMS>
       <Trace ffdc=true ffdcDirectory="c:\client\ffdc"/>
</IBM.XMS>
```

Puede inhabilitar el rastreo estableciendo FFDC en false en la sección Rastreo del archivo de configuración de aplicación.

Si no está utilizando un archivo de configuración de aplicación, FFDC está activada y el rastreo está desactivado.

Consejos para la resolución de problemas

Utilice estos consejos para resolver problemas con el uso de XMS.

La aplicación Un XMS no se puede conectar con un gestor de colas (MQRC_NOT_AUTHORIZED)

El cliente de XMS para .NET puede tener un comportamiento diferente que el comportamiento del cliente JMS de IBM WebSphere MQ. Por lo tanto, puede encontrar que la aplicación XMS no puede conectar con el gestor de colas, pero sí pueda hacerlo la aplicación JMS.

- Una solución sencilla para este problema es intentar utilizar un ID de usuario que no tenga más de 12 caracteres de longitud y que esté autorizado por completo en la lista de autoridades del gestor de colas. Si esta solución no es ideal, un enfoque diferente, aunque más complejo, sería utilizar salidas de seguridad. Si necesita ayuda adicional para este problema, consulte al servicio de soporte de IBM para recibir ayuda.
- Si establece la propiedad XMSC_USERID de la fábrica de conexiones, debe coincidir con el ID de usuario y la contraseña del usuario conectado. Si no establece esta propiedad, el gestor de colas utiliza el ID de usuario del usuario conectado, de forma predeterminada.
- La autenticación de usuario para IBM WebSphere MQ se realiza utilizando los detalles del usuario conectado actualmente y no la información proporcionada en los campos XMSC.USERID y XMSC.PASSWORD. Esto se ha diseñado para mantener la coherencia con IBM WebSphere MQ. Si desea más información, consulte *Información de autenticación* en la documentación en línea del producto IBM IBM WebSphere MQ.

Conexión redireccionada al motor de mensajería

Cuando se conecta a un bus de integración de servicios WebSphere Application Server Versión 6.0.2, todas las conexiones se pueden redireccionar desde el punto final de proveedor original al motor de mensajería que elige el bus para esa conexión de cliente. Cuando hace esto, redirige siempre la conexión a un servidor de host especificado por el nombre de host, en lugar de estar especificado por una dirección IP. Por lo tanto, puede tener problemas de conexión si el nombre de host no se puede resolver.

Para conectar satisfactoriamente con el bus de integración de servicios de WebSphere Application Server Versión 6.0.2, puede ser necesario que proporcione una correlación entre los nombres de host y las direcciones IP en su máquina hosts de cliente. Por ejemplo, puede especificar la correlación en una tabla de hosts locales en la máquina host de cliente.

Una aplicación XMS que utiliza un almacenamiento dinámico de JVM más grande

La aplicación XMS .NET que envía mensajes a través de los motores de mensajería de WebSphere Application Server normalmente necesita utilizar un almacenamiento dinámico de JVM más grande que el valor predeterminado especificado. Para cambiar los valores de configuración del almacenamiento dinámico, consulte Ajuste del rendimiento de mensajería con tecnologías de integración de servicios en la documentación del producto WebSphere Application Server Versión 7.

Soporte para la autenticación de contraseña similar al telnet

El protocolo de transporte en tiempo real XMS .NET solo admite la autenticación de contraseña simple similar al telnet. El protocolo de transporte en tiempo real XMS .NET no admite la calidad de protección.

Establecimiento de valores para el tipo de propiedad doble

En una plataforma Windows de 64 bits, es posible que los métodos SetDoubleProperty () o GetDoubleProperty () no funcionen correctamente al establecer u obtener valores para el tipo de propiedad double, si los valores son menores que Double.Epsilon.

Por ejemplo, si intenta establecer un valor de 4.9E-324 para una propiedad con el tipo double, las plataformas Windows de 64 bits la consideran como si fuera 0.0. Por eso, en un entorno de mensajería

distribuida, si JMS u otra aplicación define el valor para una propiedad doble como 4.9E-324 en alguna máquina Unix o Windows de 32 bits, y XMS .NET se ejecuta en una máquina de 64 bits, el valor devuelto por GetDoubleProperty() es 0.0. Es un problema conocido de Microsoft .NET 2.0 Framework.

Referencia de Message Service Clients para .NET

Esta sección de referencia proporciona información para ayudarle con el uso de Message Service Client para .NET. Esta información le ayuda a realizar las tareas implicadas en la programación con XMS.

Interfaces .NET

En este temasección se documentan las interfaces de clase .NET y sus propiedades y métodos.

La tabla siguiente resume todas las interfaces, que se definen en el espacio de nombres `IBM.XMS`.

Tabla 27. Resumen de las interfaces de clase .NET

Interfaz	Descripción
"IBytesMessage" en la página 95	Un mensaje de bytes es un mensaje cuyo cuerpo está formado por una corriente de bytes.
"IConnection" en la página 105	Un objeto Connection representa la conexión activa de la aplicación a un servidor de mensajería.
"IConnectionFactory" en la página 108	Una aplicación utiliza una fábrica de conexiones para crear una conexión.
"IConnectionMetaData" en la página 110	Un objeto ConnectionMetaData proporciona información sobre una conexión.
"IDestination" en la página 111	Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.
"ExceptionListener" en la página 113	Una aplicación utiliza un escucha de excepción al que se le notificará de forma asíncrona un problema con una conexión.
"IllegalStateException" en la página 114	XMS lanza esta excepción si una aplicación llama a un método en un momento incorrecto o inapropiado, o si XMS no está en un estado apropiado para la operación solicitada.
"InitialContext" en la página 114	Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.
"InvalidClientIDException" en la página 116	XMS lanza esta excepción si una aplicación intenta establecer un identificador de cliente para una conexión, pero el identificador de cliente no es válido o ya está siendo utilizado.
"InvalidDestinationException" en la página 117	XMS lanza su excepción si una aplicación especifica un destino que no es válido.
"InvalidSelectorException" en la página 117	XMS lanza esta excepción si una aplicación proporciona una expresión de selector de mensaje cuya sintaxis no es válida.

Tabla 27. Resumen de las interfaces de clase .NET (continuación)	
Interfaz	**Descripción**
“IMapMessage” en la página 117	Un mensaje de correlación es un mensaje cuyo cuerpo está formado por un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.
“IMessage” en la página 127	Un objeto Message representa un mensaje que una aplicación envía o recibe. IMessage es una superclase para las clases de mensaje como, por ejemplo, IMapMessage.
“IMessageConsumer” en la página 133	Una aplicación utiliza un consumidor de mensaje para recibir mensajes enviados a un destino.
“MessageEOFException” en la página 136	XMS lanza esta excepción si XMS encuentra el final de una corriente de mensaje de bytes cuando una aplicación lee el cuerpo de un mensaje de bytes.
“MessageFormatException” en la página 136	XMS lanza esta excepción si XMS encuentra un mensaje con un formato que no es válido.
“IMessageListener (delegado)” en la página 137	Una aplicación utiliza un escucha de mensajes para recibir mensajes de forma asíncrona.
“MessageNotReadableException” en la página 137	XMS lanza esta excepción si una aplicación intenta leer el cuerpo de un mensaje que es de solo escritura.
“MessageNotWritableException” en la página 137	XMS lanza esta excepción si una aplicación intenta escribir en el cuerpo de un mensaje que es de solo lectura.
“IMessageProducer” en la página 138	Una aplicación utiliza un productor de mensajes para enviar mensajes a un destino.
“IObjectMessage” en la página 144	Un mensaje de objeto es un mensaje cuyo cuerpo consta de un objeto Java o .NET serializado.
“IPropertyContext” en la página 145	IPropertyContext es una superclase abstracta que contiene métodos que obtienen y establecen propiedades. Estos métodos son heredados por otras clases.
“IQueueBrowser” en la página 154	Una aplicación utiliza un examinador de colas para examinar mensajes en una cola sin eliminarlas.
“Solicitante” en la página 156	Una aplicación utiliza un solicitante para enviar un mensaje de solicitud y, después, esperar una respuesta y recibirla.
“ResourceAllocationException” en la página 157	XMS lanza esta excepción si XMS no puede asignar los recursos necesarios para un método.
“SecurityException” en la página 158	XMS lanza esta excepción si se rechazan el identificador de usuario y la contraseña proporcionados para autenticar una aplicación. XMS también lanza esta excepción si una comprobación de autoridad falla e impide que se complete un método.

Tabla 27. Resumen de las interfaces de clase .NET (continuación)	
Interfaz	**Descripción**
“ISession” en la página 158	Una sesión es un contexto de hebra única para enviar y recibir mensajes.
“IStreamMessage” en la página 169	Un mensaje de corriente de datos es un mensaje cuyo cuerpo está formado por una corriente de valores, donde cada valor tiene un tipo de datos asociado.
“ITextMessage” en la página 179	Un mensaje de texto es un mensaje cuyo cuerpo está formado por una serie.
“TransactionInProgressException” en la página 180	XMS lanza esta excepción si una aplicación solicita una operación que no es válida porque una transacción está en curso.
“TransactionRolledBackException” en la página 180	XMS lanza esta excepción si una aplicación llama a Session.commit() para confirmar la transacción actual, pero la transacción se ha retrotraído.
XMSC	Para .NET, los valores y nombres de propiedad XMS se definen en esta clase como constantes públicas. Si desea más detalles, consulte “Propiedades de objetos XMS” en la página 183.
“XMSException” en la página 180	Si XMS detecta un error al procesar una llamada a un método .NET, XMS lanza una excepción. Una excepción es un objeto que encapsula información sobre el error. Existen distintos tipos de excepción XMS, y un objeto XMSException es simplemente un tipo de excepción. Sin embargo, la clase XMSException es una superclase de las otras clases de excepción XMS. XMS lanza un objeto XMSException en situaciones donde ninguno de los otros tipos de excepción es apropiado.
“XMSFactoryFactory” en la página 181	Si una aplicación no está utilizando objetos administrados, utilice esta clase para crear fábricas de conexiones, colas y temas.

La definición de cada método lista los códigos de excepción que podría devolver XMS si detecta un error al procesar una llamada al método. Cada código de excepción se representa mediante su constante con nombre, que tiene una excepción correspondiente.

Conceptos relacionados

Creación de sus propias aplicaciones
Cree sus propias aplicaciones como crea las aplicaciones de ejemplo.

Cómo escribir aplicaciones de XMS
Este secciónCapítulo proporciona información para ayudarle a escribir aplicaciones XMS.

Escritura de aplicaciones XMS para .NET
Este secciónCapítulo proporciona información para ayudarle a escribir aplicaciones XMS para .NET.

Referencia relacionada

Propiedades de objetos XMS
Este secciónCapítulo documenta las propiedades de objeto definidas por XMS.

IBytesMessage

Un mensaje de bytes es un mensaje cuyo cuerpo está formado por una corriente de bytes.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IBytesMessage
```

Referencia relacionada

Mensajes de bytes

El cuerpo de un mensaje de bytes contiene una corriente de bytes. El cuerpo solo contiene los datos reales, y es responsabilidad de las aplicaciones emisoras y receptoras interpretar estos datos.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
BodyLength	Obtener la longitud del cuerpo del mensaje en bytes cuando el cuerpo del mensaje es de solo lectura.

BodyLength - Obtener longitud de cuerpo

Interfaz:

```
Int64 BodyLength
    {
      get;
    }
```

Obtener la longitud del cuerpo del mensaje en bytes cuando el cuerpo del mensaje es de solo lectura.

El valor devuelto es la longitud de todo el cuerpo independientemente de la posición actual del cursor para leer el mensaje.

Excepciones:

- XMSException
- MessageNotReadableException

Métodos

Resumen de métodos:

Método	Descripción
ReadBoolean	Leer un valor booleano de la corriente de datos del mensaje de bytes.
ReadSignedByte	Leer el siguiente byte de la corriente de datos del mensaje de bytes como un entero de 8-bits firmado.
ReadBytes	Leer una matriz de bytes de la corriente de datos del mensaje de bytes empezando por la posición actual del cursor.
ReadChar	Leer los 2 bytes siguientes de la corriente de datos del mensaje de bytes como un carácter.
ReadDouble	Leer los 8 bytes siguiente de la corriente de datos del mensaje de bytes como un número de coma flotante de precisión doble.
ReadFloat	Leer los 4 bytes siguientes de la corriente de datos del mensaje de bytes como un número de coma flotante.

Método	Descripción
ReadInt	Leer los 4 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 32-bits firmado.
ReadLong	Leer los 8 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 64-bits firmado.
ReadShort	Leer los 2 bytes siguientes de la corriente de datos del mensaje de bytes como un entero de 16-bits firmado.
ReadByte	Leer el siguiente byte de la corriente de datos del mensaje de bytes como un entero de 8-bits sin firmar.
ReadUnsignedShort	Leer los 2 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 16-bits sin firmar.
ReadUTF	Leer una serie, codificada en UTF-8, de la corriente de datos del mensaje de bytes.
Reset	Colocar el cuerpo del mensaje en la modalidad de solo lectura y volver a colocar el cursor al inicio de la corriente de datos del mensaje de bytes.
WriteBoolean	Escribir un valor booleano en la corriente de datos de mensaje de bytes.
WriteByte	Escribir un byte en la corriente de datos del mensaje de bytes.
WriteBytes	Escribir una matriz de bytes en la corriente de datos del mensaje de bytes.
WriteBytes	Escribir una matriz de bytes parcial en la corriente de datos del mensaje de bytes, tal como lo ha definido la longitud especificada.
WriteChar	Escribir un carácter en la corriente de datos del mensaje de bytes como 2 bytes, primero el byte de orden superior.
WriteDouble	Convertir un número de coma flotante de precisión doble a un entero largo y escribir el entero largo en la corriente de datos del mensaje de bytes como 8 bytes, primero el byte de orden superior.
WriteFloat	Convertir un número de coma flotante a un entero y escribir el entero en la corriente de datos del mensaje de bytes como 4 bytes, primero el byte de orden superior.
WriteInt	Escribir un entero en la corriente de datos del mensaje de bytes como 4 bytes, primero el byte de orden superior.
WriteLong	Escribir un entero largo en la corriente de datos del mensaje de bytes como 8 bytes, primero el byte de orden superior.
WriteObject	Escribir el objeto especificado en la corriente de datos del mensaje de bytes.
WriteShort	Escribir un entero corto en la corriente de datos del mensaje de bytes como 2 bytes, primero el byte de orden superior.
WriteUTF	Escribir una serie, codificada en UTF-8, en la corriente de datos del mensaje de bytes.

ReadBoolean - Leer valor booleano

Interfaz:

```
Boolean ReadBoolean();
```

Leer un valor booleano de la corriente de datos del mensaje de bytes.

Parámetros:
Ninguna

Devuelve:
El valor booleano que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadSignedByte - Leer byte

Interfaz:

```
Int16   ReadSignedByte();
```

Leer el siguiente byte de la corriente de datos del mensaje de bytes como un entero de 8-bits firmado.

Parámetros:
Ninguna

Devuelve:
El byte que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes - Leer bytes

Interfaz:

```
Int32   ReadBytes(Byte[] array);
Int32   ReadBytes(Byte[] array, Int32 length);
```

Leer una matriz de bytes de la corriente de datos del mensaje de bytes empezando por la posición actual del cursor.

Parámetros:

array (salida)
El almacenamiento intermedio que va a contener la matriz de bytes que se ha leído. Si el número de bytes que faltan por leer de la corriente de datos antes de la llamada es mayor o igual que la longitud del almacenamiento intermedio, el almacenamiento intermedio se rellena. De lo contrario, el almacenamiento intermedio se rellena de forma parcial con todos los bytes restantes.

Si especifica un puntero nulo en la entrada, el método se salta los bytes sin leerlos. Si el número de bytes que faltan por leer de la corriente de datos antes de la llamada es mayor o igual que la longitud del almacenamiento intermedio, el número de bytes omitidos es igual a la longitud del almacenamiento intermedio. De lo contrario, se omiten todos los bytes restantes. El cursor se deja en la siguiente posición para leer en la corriente de datos del mensaje de bytes.

length (entrada)
La longitud del almacenamiento intermedio en bytes

Devuelve:
El número de bytes que se van a leer en el almacenamiento intermedio. Si el almacenamiento intermedio se rellena de forma parcial, el valor es menor que la longitud del almacenamiento intermedio, que indica que no queda ningún byte más para leer. Si no queda ningún bytes para leer en la corriente de datos antes de la llamada, el valor es XMSC_END_OF_STREAM.

Si especifica un puntero nulo en la entrada, el método no devuelve ningún valor.

Excepciones:

- XMSException
- MessageNotReadableException

ReadChar - Leer carácter

Interfaz:

```
Char    ReadChar();
```

Leer los 2 bytes siguientes de la corriente de datos del mensaje de bytes como un carácter.

Parámetros:
Ninguna

Devuelve:
El carácter que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble - Leer número de coma flotante de precisión doble

Interfaz:

```
Double  ReadDouble();
```

Leer los 8 bytes siguiente de la corriente de datos del mensaje de bytes como un número de coma flotante de precisión doble.

Parámetros:
Ninguna

Devuelve:
El número de coma flotante de precisión doble que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat - Leer número de coma flotante

Interfaz:

```
Single  ReadFloat();
```

Leer los 4 bytes siguientes de la corriente de datos del mensaje de bytes como un número de coma flotante.

Parámetros:
Ninguna

Devuelve:
El número de coma flotante que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt - Leer entero

Interfaz:

```
Int32   ReadInt();
```

Leer los 4 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 32-bits firmado.

Parámetros:
Ninguna

Devuelve:
El entero que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong - Leer entero largo

Interfaz:

```
Int64   ReadLong();
```

Leer los 8 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 64-bits firmado.

Parámetros:
Ninguna

Devuelve:
El entero largo que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadShort - Leer entero corto

Interfaz:

```
Int16   ReadShort();
```

Leer los 2 bytes siguientes de la corriente de datos del mensaje de bytes como un entero de 16-bits firmado.

Parámetros:
Ninguna

Devuelve:
El entero corto que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte - Leer byte sin firmar

Interfaz:

```
Byte    ReadByte();
```

Leer el siguiente byte de la corriente de datos del mensaje de bytes como un entero de 8-bits sin firmar.

Parámetros:
Ninguna

Devuelve:
El byte que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadUnsignedShort - Leer entero corto sin firmar

Interfaz:

```
Int32   ReadUnsignedShort();
```

Leer los 2 bytes siguiente de la corriente de datos del mensaje de bytes como un entero de 16-bits sin firmar.

Parámetros:
Ninguna

Devuelve:
El entero corto sin firma que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadUTF - Leer serie UTF

Interfaz:

```
String  ReadUTF();
```

Leer una serie, codificada en UTF-8, de la corriente de datos del mensaje de bytes.

Nota: Antes de llamar a ReadUTF(), asegúrese de que el cursor del almacenamiento intermedio está apuntado al inicio de la corriente de datos del mensaje de bytes.

Parámetros:
Ninguna

Devuelve:
Un objeto String que encapsula la serie que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

Reset - Restablecer

Interfaz:

```
void Reset();
```

Colocar el cuerpo del mensaje en la modalidad de solo lectura y volver a colocar el cursor al inicio de la corriente de datos del mensaje de bytes.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotReadableException

WriteBoolean - Escribir valor booleano

Interfaz:

```
void  WriteBoolean(Boolean value);
```

Escribir un valor booleano en la corriente de datos de mensaje de bytes.

Parámetros:

value (entrada)
El valor booleano que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteByte - Escribir byte

Interfaz:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Escribir un byte en la corriente de datos del mensaje de bytes.

Parámetros:

value (entrada)
El byte que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteBytes - Escribir bytes

Interfaz:

```
void  WriteBytes(Byte[] value);
```

Escribir una matriz de bytes en la corriente de datos del mensaje de bytes.

Parámetros:

value (entrada)
La matriz de bytes que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteBytes - Escribir matriz de bytes parcial

Interfaz:

```
void  WriteBytes(Byte[] value, int offset, int length);
```

Escribir una matriz de bytes parcial en la corriente de datos del mensaje de bytes, tal como lo ha definido la longitud especificada.

Parámetros:

value (entrada)
La matriz de bytes que se va a escribir.

offset (entrada)
El punto de inicio para la matriz de bytes que se va a escribir.

length (entrada)
El número de bytes que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteChar - Escribir carácter

Interfaz:

```
void  WriteChar(Char value);
```

Escribir un carácter en la corriente de datos del mensaje de bytes como 2 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El carácter que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteDouble - Escribir número de coma flotante de precisión doble

Interfaz:

```
void  WriteDouble(Double value);
```

Convertir un número de coma flotante de precisión doble a un entero largo y escribir el entero largo en la corriente de datos del mensaje de bytes como 8 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El número de coma flotante de precisión doble que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteFloat - Escribir número de coma flotante

Interfaz:

```
void  WriteFloat(Single value);
```

Convertir un número de coma flotante a un entero y escribir el entero en la corriente de datos del mensaje de bytes como 4 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El número de coma flotante que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteInt - Escribir entero

Interfaz:

```
void  WriteInt(Int32 value);
```

Escribir un entero en la corriente de datos del mensaje de bytes como 4 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteLong - Escribir entero largo

Interfaz:

```
void  WriteLong(Int64 value);
```

Escribir un entero largo en la corriente de datos del mensaje de bytes como 8 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero largo que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteObject - Escribir objeto

Interfaz:

```
void  WriteObject(Object value);
```

Escribir el objeto especificado en la corriente de datos del mensaje de bytes.

Parámetros:

value (entrada)
El objeto que se va a escribir, que debe hacer referencia a u n tipo primitivo.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteShort - Escribir entero corto

Interfaz:

```
void  WriteShort(Int16 value);
```

Escribir un entero corto en la corriente de datos del mensaje de bytes como 2 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero corto que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteUTF - Escribir serie UTF

Interfaz:

```
void  WriteUTF(String value);
```

Escribir una serie, codificada en UTF-8, en la corriente de datos del mensaje de bytes.

Parámetros:

value (entrada)
Un objeto String que encapsula la serie que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

Propiedades y métodos heredados

Las propiedades siguientes se heredan de la interfaz IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Los métodos siguientes se heredan de la interfaz IMessage:

clearBody, clearProperties, PropertyExists

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IConnection

Un objeto Connection representa la conexión activa de la aplicación a un servidor de mensajería.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnection
```

Si desea una lista de las propiedades definidas de XMS de un objeto Connection, consulte “Propiedades de conexión” en la página 184.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
ClientID	Obtener y establecer el identificador de cliente para la conexión.
ExceptionListener	Obtener el escucha de excepción que está registro con la conexión y registrar un escucha de excepción con la conexión.
MetaData	Obtener los metadatos para la conexión.

ClientID - Obtener y establecer ID de cliente

Interfaz:

```
String ClientID
      {
        get;
        set;
      }
```

Obtener y establecer el identificador de cliente para la conexión.

El administrador puede preconfigurar el identificador de cliente en objeto ConnectionFactory, o asignarlo estableciendo ClientID.

Un identificador de cliente se utiliza solo para dar soporte a suscripciones duraderas en el dominio de publicación/suscripción y se ignora en el dominio punto a punto.

Si una aplicación establece un identificador de cliente para una conexión, la aplicación debe hacerlo inmediatamente después de crear la conexión y antes de realizar cualquier otra operación en la conexión. Si la aplicación intenta establecer un identificador de cliente después de este punto, la llamada lanza la excepción IllegalStateException.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

Excepciones:

- XMSException
- IllegalStateException
- InvalidClientIDException

ExceptionListener - Obtener y establecer escucha de excepción

Interfaz:

```
    ExceptionListener ExceptionListener
        {
          get;
          set;
        }
```

Obtener el escucha de excepción que está registro con la conexión y registrar un escucha de excepción con la conexión.

Si no hay registrado ningún escucha de excepción con la conexión, el método devuelve un valor nulo. Si ya hay un escucha de excepción registrado con la conexión, puede cancelar el registro especificando un valor nulo, en lugar del escucha de excepción.

Si desea más información sobre cómo utilizar el escucha de excepción, consulte “Escuchas de mensajes y excepción en .NET” en la página 50.

Excepciones:

- XMSException

Metadata - Obtener metadatos

Interfaz:

```
IConnectionMetaData MetaData
    {
      get;
    }
```

Obtener los metadatos para la conexión.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar la conexión.
CreateSession	Crear una sesión.
Start	Iniciar o reiniciar la entrega de mensajes entrantes para la conexión.
Stop	Detener la entrega de mensajes entrantes para la conexión.

Close - Cerrar conexión

Interfaz:

```
void  Close();
```

Cerrar la conexión.

Si una aplicación intenta cerrar una conexión que ya está cerrar, la llamada se ignora.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

CreateSession - Crear sesión

Interfaz:

```
    ISession CreateSession(Boolean transacted,
                           AcknowledgeMode acknowledgeMode);
```

Crear una sesión.

Parámetros:

transacted (entrada)
El valor `True` significa que la sesión es una sesión con transacción. El valor `False` significa que la sesión no es una sesión con transacción.

Para una conexión en tiempo real con un intermediario, el valor debe ser `False`.

acknowledgeMode (entrada)
Indica cómo se acusa el recibo de los mensajes recibidos por una aplicación. El valor debe adoptar uno de los valores siguientes del enumerador AcknowledgeMode:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Para una conexión en tiempo real con un intermediario, el valor debe ser `AcknowledgeMode.AutoAcknowledge` o `AcknowledgeMode.DupsOkAcknowledge`

Este parámetro se ignora si la sesión es una sesión con transacción. Si desea más información sobre las modalidades de acuse de recibo, consulte “Acuse de recibo de mensaje” en la página 28.

Devuelve:
El objeto Session

Excepciones:

- XMSException

Start - Iniciar conexión

Interfaz:

```
void  Start();
```

Iniciar o reiniciar la entrega de mensajes entrantes para la conexión. La llamada se ignora si la conexión ya se ha iniciado.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

Stop - Detener conexión

Interfaz:

```
void  Stop();
```

Detener la entrega de mensajes entrantes para la conexión. La llamada se ignora, si la conexión ya se ha detenido.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IConnectionFactory

Una aplicación utiliza una fábrica de conexiones para crear una conexión.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionFactory
```

Si desea una lista de las propiedades definidas de XMS de un objeto ConnectionFactory, consulte “Propiedades de ConnectionFactory” en la página 185.

Conceptos relacionados
Objetos ConnectionFactories y Connection

Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Conexiones seguras a un IBM WebSphere MQ gestor de colas
Para habilitar una aplicación XMS .NET para realizar conexiones seguras con un IBM WebSphere MQ gestor de colas, las propiedades relevantes deben estar definidas en el objeto ConnectionFactory .

Conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus
Para permitir que una aplicación XMS de .NET establezca conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus, se deben definir las propiedades pertinentes en el objeto ConnectionFactory.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Métodos

Resumen de métodos:

Método	Descripción
CreateConnection	Crear una fábrica de conexiones con las propiedades predeterminadas.
CreateConnection	Crear una conexión utilizando una identidad de usuario especificada.

CreateConnection - Crear fábrica de conexiones (utilizando la identidad de usuario predeterminada)

Interfaz:

```
IConnection CreateConnection();
```

Crear una fábrica de conexiones con las propiedades predeterminadas.

Si está conectándose a IBM WebSphere MQ, y establece la propiedadXMSC_USERID de la fábrica de conexiones, debe coincidir con el **userid** del usuario conectado. Si no establece estas propiedades, el gestor de colas utiliza el **userid** del usuario conectado de forma predeterminada. Si necesita más autenticación a nivel de conexión de usuarios individuales, puede escribir una salida de autenticación de cliente que esté configurada en IBM WebSphere MQ.

Parámetros:
Ninguna

Excepciones:

- XMSException

CreateConnection - Crear conexión (utilizando una identidad de usuario especificada)

Interfaz:

```
IConnection CreateConnection(String userId, String password);
```

Crear una conexión utilizando una identidad de usuario especificada.

Si está conectándose a IBM WebSphere MQ, y establece la propiedadXMSC_USERID de la fábrica de conexiones, debe coincidir con el **userid** del usuario conectado. Si no establece estas propiedades, el gestor de colas utiliza el **userid** del usuario conectado de forma predeterminada. Si necesita más autenticación a nivel de conexión de usuarios individuales, puede escribir una salida de autenticación de cliente que esté configurada en IBM WebSphere MQ.

La conexión se crea en la modalidad detenida. No se entrega ningún mensaje hasta que la aplicación llama a **Connection.start()**.

Parámetros:

userID (entrada)
Un objeto String que encapsula el identificador de usuario que se va a utilizar para autenticar la aplicación. Si proporciona un valor nulo, se realiza un intento de crear la conexión sin autenticación.

password (entrada)
Un objeto String que encapsula la contraseña que se va a utilizar para autenticar la aplicación. Si proporciona un valor nulo, se realiza un intento de crear la conexión sin autenticación.

Devuelve:
El objeto Connection.

Excepciones:

- XMSException
- XMS_X_SECURITY_EXCEPTION

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IConnectionMetaData

Un objeto ConnectionMetaData proporciona información sobre una conexión.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionMetaData
```

Si desea una lista de las propiedades definidas de XMS de un objeto ConnectionMetaData, consulte “Propiedades de ConnectionMetaData” en la página 191.

Propiedades de .NET

Resumen de propiedades

Método	Descripción
JMSXPropertyNames	Devolver una enumeración de los nombres de las propiedades de mensaje definidas por JMS soportadas por la conexión.

JMSXPropertyNames - Obtener propiedades de mensaje definidas de JMS

Interfaz:

```
System.Collections.IEnumerator JMSXPropertyNames
    {
      get;
    }
```

Devolver una enumeración de los nombres de las propiedades de mensaje definidas por JMS soportadas por la conexión.

Las propiedades de mensaje definidas de JMS no están soportadas por una conexión en tiempo real con un intermediario.

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IDestination

Un destino es el lugar al que una aplicación envía mensajes, o es un origen del cual una aplicación recibe mensajes, o ambas cosas.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IDestination
```

Si desea una lista de las propiedades definidas de XMS de un objeto Destination, consulte “Propiedades de Destination” en la página 191.

Conceptos relacionados

Objetos ConnectionFactories y Connection
Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Destinos
Una aplicación XMS utiliza un objeto Destination para especificar el destino de mensajes que se están enviando y el origen de mensajes que se están recibiendo.

Comodines de destino

XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Identificadores uniformes de recursos de tema
El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

Identificadores uniformes de recursos de cola
El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

Destinos temporales
Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

Propiedades de .NET

Resumen de métodos:

Método	Descripción
Nombre	Obtener el nombre del destino.
TypeId	Obtener el tipo del destino.

Name - Obtener nombre de destino

Interfaz:

```
String Name
{
  get;
}
```

Obtener el nombre del destino. El nombre es una serie que encapsula el nombre de una cola, o bien el nombre de un tema.

Excepciones:

- XMSException

TypeId - Obtener tipo de destino

Interfaz:

```
DestinationType TypeId
{
  get;
}
```

Obtener el tipo del destino. El tipo del destino es uno de los valores siguientes:

```
DestinationType.Queue
DestinationType.Topic
```

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

ExceptionListener

Jerarquía de herencia:
Ninguna

Una aplicación utiliza un escucha de excepción al que se le notificará de forma asíncrona un problema con una conexión.

Si una aplicación utiliza una conexión solo para consumir mensajes de forma asíncrona, y sin ninguna otra finalidad, la única forma a través de la cual la aplicación puede obtener información sobre un problema con la conexión es utilizar un escucha de excepción. En otras situaciones, un escucha de excepción puede proporcionar una forma más inmediata de obtener información sobre un problema con una conexión que esperar hasta la siguiente llamada síncrona a XMS.

Delegado

Resumen de delegado

Delegado	Descripción
ExceptionListener	Notifique a la aplicación un problema con una conexión.

ExceptionListener - Escucha de excepción

Interfaz:

```
public delegate void ExceptionListener(Exception ex)
```

Notifique a la aplicación un problema con una conexión.

Los métodos que implementan este delegado se pueden registrar con la conexión.

Si desea más información sobre cómo utilizar el escucha de excepción, consulte “Escuchas de mensajes y excepción en .NET” en la página 50.

Parámetros:

exception (entrada)
Un puntero para una excepción creada por XMS.

Devuelve:
Void

IllegalStateException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.Exception
           |
           +----IBM.XMS.IllegalStateException
```

XMS lanza esta excepción si una aplicación llama a un método en un momento incorrecto o inapropiado, o si XMS no está en un estado apropiado para la operación solicitada.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

InitialContext

Una aplicación utiliza un objeto InitialContext para crear objetos a partir de definiciones de objeto que se han recuperado de un repositorio de objetos administrados.

Jerarquía de herencia:
Ninguna

Conceptos relacionados

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
Environment	Obtener el entorno.

Environment - Obtener el entorno

Interfaz:

```
Hashtable Environment
{
     get;
}
```

Obtener el entorno.

Excepciones:

- Las excepciones son específicas al servicio de directorio que se está utilizando.

Constructores

Resumen de constructores

Constructor	Descripción
InitialContext	Crear un objeto InitialContext.

InitialContext - Crear contexto inicial

Interfaz:

```
InitialContext(Hashtable env);
```

Crear un objeto InitialContext.

Parámetros:

La información necesaria para establecer una conexión al repositorio de objetos administrados se proporciona al constructor en una tabla hash de entorno.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
AddToEnvironment	Añadir una nueva propiedad al entorno.
Cerrar	Cerrar este contexto.
Lookup	Crear un objeto a partir de una definición de objeto que se ha recuperado del repositorio de objetos administrados.
RemoveFromEnvironment	Eliminar una propiedad del entorno.

AddToEnvironment - Añadir una nueva propiedad al entorno

Interfaz:

```
Object AddToEnvironment(String propName, Object propVal);
```

Añadir una nueva propiedad al entorno.

Parámetros:

propName (entrada
Un objeto String que encapsula el nombre de la propiedad que se va a añadir.

propVal (entrada)
El valor de la propiedad que se va añadir.

Devuelve:
El valor antiguo de la propiedad.

Excepciones:

- Las excepciones son específicas al servicio de directorio que se está utilizando.

Close - Cerrar este contexto

Interfaz:

```
void Close()
```

Cerrar este contexto.

Parámetros:
Ninguna

Devuelve:
Ninguna

Excepciones:

- Las excepciones son específicas al servicio de directorio que se está utilizando.

Lookup - Buscar objeto en contexto inicial

Interfaz:

```
Object Lookup(String name);
```

Crear un objeto a partir de una definición de objeto que se ha recuperado del repositorio de objetos administrados.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre del objeto administrado que se va a recuperar. El nombre puede ser un nombre sencillo o un nombre complejo. Si desea más detalles, consulte “Recuperación de objetos administrados” en la página 66.

Devuelve:
IConnectionFactory o IDestination, en función del tipo de objeto que se está recuperando. Si la función puede acceder al directorio, pero no puede encontrar el objeto necesario, se devuelve un valor nulo.

Excepciones:

- Las excepciones son específicas al servicio de directorio que se está utilizando.

RemoveFromEnvironment - Eliminar una propiedad del entorno

Interfaz:

```
Object RemoveFromEnvironment(String propName);
```

Eliminar una propiedad del entorno.

Parámetros:

propName (entrada
Un objeto String que encapsula el nombre de la propiedad que se va a eliminar.

Devuelve:
El objeto que se ha eliminado.

Excepciones:

- Las excepciones son específicas al servicio de directorio que se está utilizando.

InvalidClientIDException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
```

```
          |
          +----IBM.XMS.InvalidClientIDException
```

XMS lanza esta excepción si una aplicación intenta establecer un identificador de cliente para una conexión, pero el identificador de cliente no es válido o ya está siendo utilizado.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

InvalidDestinationException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidDestinationException
```

XMS lanza su excepción si una aplicación especifica un destino que no es válido.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

InvalidSelectorException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidSelectorException
```

XMS lanza esta excepción si una aplicación proporciona una expresión de selector de mensaje cuya sintaxis no es válida.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

IMapMessage

Un mensaje de correlación es un mensaje cuyo cuerpo está formado por un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IMapMessage
```

Cuando una aplicación obtiene el valor del par nombre-valor, el valor se puede convertir mediante XMS a otro tipo de datos. Si desea más información sobre esta forma de conversión implícita, consulte “Mensajes de correlación” en la página 80.

Referencia relacionada

Mensajes de correlación

El cuerpo de un mensaje de correlación contiene un conjunto de pares de nombre-valor, donde cada valor tiene un tipo de datos asociado.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
MapNames	Obtener una enumeración de los nombres en el cuerpo del mensaje de correlación.

MapNames - Obtener nombres de correlación

Interfaz:

```
System.Collections.IEnumerator MapNames
{
  get;
}
```

Obtener una enumeración de los nombres en el cuerpo del mensaje de correlación.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
GetBoolean	Obtener el valor booleano identificado por el nombre del cuerpo del mensaje de correlación.
GetByte	Obtener el byte identificado por el nombre del cuerpo del mensaje de correlación.
GetBytes	Obtener la matriz de bytes identificada por el nombre del cuerpo del mensaje de correlación.
GetChar	Obtener el carácter identificado por el nombre del cuerpo del mensaje de correlación.
GetDouble	Obtener el número de coma flotante de precisión doble identificado por el nombre del cuerpo del mensaje de correlación.
GetFloat	Obtener el número de coma flotante identificado por el nombre del cuerpo del mensaje de correlación.
GetInt	Obtener el entero identificado por el nombre del cuerpo del mensaje de correlación.
GetLong	Obtener el entero largo identificado por el nombre del cuerpo del mensaje de correlación.
GetObject	Obtener una referencia al valor de un par de nombre-valor del cuerpo del mensaje de correlación.
GetShort	Obtener el entero corto identificado por el nombre del cuerpo del mensaje de correlación.
GetString	Obtener la serie identificada por el nombre del cuerpo del mensaje de correlación.

Método	Descripción
ItemExists	Comprobar si el cuerpo del mensaje de correlación contiene un par nombre-valor con el nombre especificado.
SetBoolean	Establecer un valor booleano en el cuerpo del mensaje de correlación.
SetByte	Establecer un byte en el cuerpo del mensaje de correlación.
SetBytes	Establecer una matriz debytes en el cuerpo del mensaje de correlación.
SetChar	Establecer un carácter de 2-bytes en el cuerpo del mensaje de correlación.
SetDouble	Establecer un número de coma flotante de precisión doble en el cuerpo del mensaje de correlación.
SetFloat	Establecer un número de coma flotante en el cuerpo del mensaje de correlación.
SetInt	Establecer un entero en el cuerpo del mensaje de correlación.
SetLong	Establecer un entero largo en el cuerpo del mensaje de correlación.
SetObject	Establecer un valor, que debe ser un tipo primitivo XMS, en el cuerpo del mensaje de correlación.
SetShort	Establecer un entero corto en el cuerpo del mensaje de correlación.
SetString	Establecer una serie en el cuerpo del mensaje de correlación.

GetBoolean - Obtener valor booleano

Interfaz:

```
Boolean GetBoolean(String name);
```

Obtener el valor booleano identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el valor booleano.

Devuelve:
El valor booleano recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetByte - Obtener byte

Interfaz:

```
Byte    GetByte(String name);
    Int16   GetSignedByte(String name);
```

Obtener el byte identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el byte.

Devuelve:
El byte recuperado del cuerpo del mensaje de correlación. No se realiza ninguna conversión de datos en el byte.

Excepciones:

- XMSException

GetBytes - Obtener bytes

Interfaz:

```
Byte[]  GetBytes(String name);
```

Obtener la matriz de bytes identificada por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica la matriz de bytes.

Devuelve:
El número de bytes de la matriz.

Excepciones:

- XMSException

GetChar - Obtener carácter

Interfaz:

```
Char    GetChar(String name);
```

Obtener el carácter identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (input)
Un objeto String que encapsula el nombre que identifica el carácter.

Devuelve:
El carácter recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetDouble - Obtener número de coma flotante de precisión doble

Interfaz:

```
Double  GetDouble(String name);
```

Obtener el número de coma flotante de precisión doble identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el número de coma flotante de precisión doble.

Devuelve:
El número de coma flotante de precisión doble recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetFloat - Obtener número de coma flotante

Interfaz:

```
Single  GetFloat(String name);
```

Obtener el número de coma flotante identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el número de coma flotante.

Devuelve:
El número de coma flotante recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetInt - Obtener entero

Interfaz:

```
Int32   GetInt(String name);
```

Obtener el entero identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el entero.

Devuelve:
El entero recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetLong - Obtener entero largo

Interfaz:

```
Int64   GetLong(String name);
```

Obtener el entero largo identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el entero largo.

Devuelve:
El entero largo recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetObject - Obtener objeto

Interfaz:

```
Object  GetObject(String name);
```

Obtener una referencia al valor de un par de nombre-valor del cuerpo del mensaje de correlación. El par nombre-valor se identifica por el nombre.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre del par nombre-valor.

Devuelve:
El valor, que es uno de los tipos de objeto siguientes:

```
Boolean
Byte
```

Byte[]
Char
Double
Single
Int32
Int64
Int16
String

Excepciones:
XMSException

GetShort - Obtener entero corto

Interfaz:

```
Int16   GetShort(String name);
```

Obtener el entero corto identificado por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica el entero corto.

Devuelve:
El entero corto recuperado del cuerpo del mensaje de correlación.

Excepciones:

- XMSException

GetString - Obtener serie

Interfaz:

```
String  GetString(String name);
```

Obtener la serie identificada por el nombre del cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre que identifica la serie en el cuerpo del mensaje de correlación.

Devuelve:
Un objeto String que encapsula la serie recuperada del cuerpo del mensaje de correlación. Si es necesaria una conversión de datos, este valor es la serie después de la conversión.

Excepciones:

- XMSException

ItemExists - Comprobar si existe el par nombre-valor

Interfaz:

```
Boolean ItemExists(String name);
```

Comprobar si el cuerpo del mensaje de correlación contiene un par nombre-valor con el nombre especificado.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre del par nombre-valor.

Devuelve:

- `True`, si el cuerpo del mensaje de correlación contiene un par nombre-valor con el nombre especificado.
- `False`, si el cuerpo del mensaje de correlación no contiene un par nombre-valor con el nombre especificado.

Excepciones:

- XMSException

SetBoolean - Establecer valor booleano

Interfaz:

```
void  SetBoolean(String name, Boolean value);
```

Establecer un valor booleano en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el valor booleano en el cuerpo del mensaje de correlación.

value (entrada)
El valor booleano que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetByte - Establecer byte

Interfaz:

```
void  SetByte(String name, Byte value);
void  SetSignedByte(String name, Int16 value);
```

Establecer un byte en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el byte en el cuerpo del mensaje de correlación.

value (entrada)
El byte que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetBytes - Establecer bytes

Interfaz:

```
void  SetBytes(String name, Byte[] value);
```

Establecer una matriz debytes en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar la matriz de bytes en el cuerpo del mensaje de correlación.

value (entrada)
La matriz de bytes que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetChar - Establecer carácter

Interfaz:

```
void  SetChar(String name, Char value);
```

Establecer un carácter de 2-bytes en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el carácter en el cuerpo del mensaje de correlación.

value (entrada)
El carácter que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetDouble - Establecer número de coma flotante de precisión doble

Interfaz:

```
void  SetDouble(String name, Double value);
```

Establecer un número de coma flotante de precisión doble en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el número de coma flotante de precisión doble en el cuerpo del mensaje de correlación.

value (entrada)
El número de coma flotante de precisión doble que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetFloat - Establecer número de coma flotante

Interfaz:

```
void  SetFloat(String name, Single value);
```

Establecer un número de coma flotante en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el número de coma flotante en el cuerpo del mensaje de correlación.

value (entrada)
El número de coma flotante que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetInt - Establecer entero

Interfaz:

```
void  SetInt(String name, Int32 value);
```

Establecer un entero en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el entero en el cuerpo del mensaje de correlación.

value (entrada)
El entero que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetLong - Establecer entero largo

Interfaz:

```
void  SetLong(String name, Int64 value);
```

Establecer un entero largo en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el entero largo en el cuerpo del mensaje de correlación.

value (entrada)
El entero largo que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetObject - Establecer objeto

Interfaz:

```
void  SetObject(String name, Object value);
```

Establecer un valor, que debe ser un tipo primitivo XMS, en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el valor en el cuerpo del mensaje de correlación.

value (entrada)
Una matriz de bytes que contiene el valor que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetShort - Establecer entero corto

Interfaz:

```
void  SetShort(String name, Int16 value);
```

Establecer un entero corto en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar el entero corto en el cuerpo del mensaje de correlación.

value (entrada)
El entero corto que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

SetString - Establecer serie

Interfaz:

```
void  SetString(String name, String value);
```

Establecer una serie en el cuerpo del mensaje de correlación.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre para identificar la serie en el cuerpo del mensaje de correlación.

value (entrada)
Un objeto String que encapsula la serie que se va a establecer.

Devuelve:
Void

Excepciones:

- XMSException

Propiedades y métodos heredados

Las propiedades siguientes se heredan de la interfaz IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Los métodos siguientes se heredan de la interfaz IMessage:

clearBody, clearProperties, PropertyExists

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IMessage

Un objeto Message representa un mensaje que una aplicación envía o recibe. IMessage es una superclase para las clases de mensaje como, por ejemplo, IMapMessage.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
```

Si desea una lista de los campos de cabecera de mensaje JMS en un objetoMessage, consulte “Campos de cabecera en mensajes de Un XMS” en la página 73. Para obtener una lista de las propiedades definidas de JMS de un objeto Message, consulte “Propiedades definidas por JMS de un mensaje” en la página 75. Si desea una lista de las propiedades definidas de IBM de un objeto Message, consulte “Propiedades definidas por IBM de un mensaje” en la página 75. Para obtener una lista de propiedades JMS_IBM_MQMD* para el objeto Message, consulte “Propiedades JMS_IBM_MQMD*” en la página 197

Los mensajes son suprimidos por el recopilador de basura. Cuando se suprime un mensaje, esta acción libera los recursos que estaba utilizando.

Referencia relacionada

Mensajes de XMS
Este secciónCapítulo describe la estructura y el contenido de los mensajes de XMS y explica cómo las aplicaciones procesan los mensajes de XMS.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
JMSCorrelationID	Obtener y establecer el identificador de correlación del mensaje como un objeto String.
JMSDeliveryMode	Obtener y establecer la modalidad de entrega del mensaje.
JMSDestination	Obtener y establecer el destino del mensaje.
JMSExpiration	Obtener y establecer la hora de caducidad del mensaje.
JMSMessageID	Obtener y establecer el identificador de mensaje del mensaje como un objeto de serie que encapsula el identificador de mensaje.
JMSPriority	Obtener y establecer la prioridad del mensaje.
JMSRedelivered	Obtener una indicación de si el mensaje se está volviendo a entregar e indicar si el mensaje se está volviendo a entregar.
JMSReplyTo	Obtener y establecer el destino adonde se va a enviar una respuesta al mensaje.
JMSTimestamp	Obtener y establecer la hora cuando se envió el mensaje.
JMSType	Obtener y establecer el tipo de mensaje.

Propiedad .NET	Descripción
PropertyNames	Obtener una enumeración de las propiedades de nombres del mensaje.

GetJMSCorrelationID - Obtener y establecer JMSCorrelationID

Interfaz:

```
String JMSCorrelationID
{
  get;
  set;
}
```

Obtener y establecer el identificador de correlación del mensaje como un objeto String.

Excepciones:

- XMSException

JMSDeliveryMode - Obtener y establecer JMSDeliveryMode

Interfaz:

```
DeliveryMode JMSDeliveryMode
{
  get;
  set;
}
```

Obtener y establecer la modalidad de entrega del mensaje.

La modalidad de entrega del mensaje adopta uno de los valores siguientes:

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

Para un mensaje recién creado que no se ha enviado, la modalidad de entrega es `DeliveryMode.Persistent`, excepto para una conexión en tiempo real a un intermediario para el cual la modalidad de entrega es `DeliveryMode.NonPersistent`. Para un mensaje que se recibe, el método devuelve la modalidad de entrega que estableció la llamada IMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie la modalidad de entrega estableciendo JMSDeliveryMode.

Excepciones:

- XMSException

JMSDestination - Obtener y establecer JMSDestination

Interfaz:

```
IDestination JMSDestination
{
  get;
  set;
}
```

Obtener y establecer el destino del mensaje.

El destino se establece mediante la llamada IMessageProducer.send() cuando se envía el mensaje. Se ignora el valor de JMSDestination. Sin embargo, puede utilizar JMSDestination para cambiar el destino de un mensaje que se ha recibido.

Para un mensaje recién creado que no se ha enviado, el método devuelve un objeto Destination nulo, a menos que la aplicación emisora establezca un destino estableciendo JMSDestination. Para un mensaje que se ha recibido, el método devuelve un objeto Destination para el destino que estableció la llamada IMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie el destino estableciendo JMSDestination.

Excepciones:

- XMSException

JMSExpiration - Obtener y establecer JMSExpiration

Interfaz:

```
Int64 JMSExpiration
{
  get;
  set;
}
```

Obtener y establecer la hora de caducidad del mensaje.

La hora de caducidad se establece mediante la llamada IMessageProducer.send() cuando se envía el mensaje. Su valor se calcula añadiendo el tiempo de vida, tal como lo especifica la aplicación emisora, a la hora cuando se envía el mensaje. La hora de caducidad se expresa en milisegundos desde las 00:00:00 GMT el 1 de enero de 1970.

Para un mensaje recién creado que no se ha enviado, la hora de caducidad es 0, a menos que la aplicación emisora defina una hora de caducidad diferente estableciendo JMSExpiration. Para un mensaje que se ha recibido, el método devuelve la hora caducidad que estableció la llamada IMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie la hora de caducidad estableciendo JMSExpiration.

Si la hora de caducidad es 0, la llamada IMessageProducer.send() establece la hora de caducidad en 0 para indicar que el mensaje no caduca.

XMS descarta mensajes caducados y no los entrega a aplicaciones.

Excepciones:

- XMSException

JMSMessageID - Obtener y establecer JMSMessageID

Interfaz:

```
String JMSMessageID
{
  get;
  set;
}
```

Obtener y establecer el identificador de mensaje del mensaje como un objeto de serie que encapsula el identificador de mensaje.

El identificador de mensaje se establece mediante la llamada IMessageProducer.send() cuando se envía el mensaje. Para un mensaje que se ha recibido, el método devuelve el identificador de mensaje que estableció la llamadaIMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie el identificador de mensaje estableciendo JMSMessageID.

Si el mensaje no tiene ningún identificador de mensaje, el método devuelve un valor nulo.

Excepciones:

- XMSException

JMSPriority - Obtener y establecer JMSPriority

Interfaz:

```
Int32 JMSPriority
{
  get;
  set;
}
```

Obtener y establecer la prioridad del mensaje.

La prioridad se establece mediante la llamada IMessageProducer.send() cuando se envía el mensaje. El valor es un entero dentro del rango de 0, la prioridad inferior, a 9, la prioridad superior.

Para un mensaje recién creado que no se ha enviado, la prioridad es 4, a menos que la aplicación emisora establezca una prioridad diferente estableciendo JMSPriority. Para un mensaje que se ha recibido, el método devuelve la prioridad que estableció la llamada IMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie la prioridad estableciendo JMSPriority.

Excepciones:

- XMSException

JMSRedelivered - Obtener y establecer JMSRedelivered

Interfaz:

```
Boolean JMSRedelivered
{
  get;
  set;
}
```

Obtener una indicación de si el mensaje se está volviendo a entregar e indicar si el mensaje se está volviendo a entregar. La indicación se establece mediante la llamada IMessageConsumer.receive() cuando se recibe el mensaje.

Esta propiedad tiene los valores siguientes:

- `True`, si el mensaje se está volviendo a entregar.
- `False`, si el mensaje no se está volviendo a entregar.

Para una conexión en tiempo real a un intermediario, el valor siempre es `False`.

Una indicación de reentrega establecida por JMSRedelivered antes de que se envíe el mensaje es ignorada por la llamada IMessageProducer.send() cuando se envía el mensaje, y la llamada IMessageConsumer.receive() la ignora y la sustituye cuando se recibe el mensaje. Sin embargo, puede utilizar JMSRedelivered para cambiar la indicación para un mensaje que se ha recibido.

Excepciones:

- XMSException

JMSReplyTo - Obtener y establecer JMSReplyTo

Interfaz:

```
IDestination JMSReplyTo
{
  get;
  set;
}
```

Obtener y establecer el destino adonde se va a enviar una respuesta al mensaje.

El valor de esta propiedad es un objeto Destination para el destino adonde se va a enviar una respuesta al mensaje. Un objeto Destination nulo significa que no se espera ninguna respuesta.

Excepciones:

- XMSException

JMSTimestamp - Obtener y establecer JMSTimestamp

Interfaz:

```
Int64 JMSTimestamp
{
  get;
```

```
  set;
}
```

Obtener y establecer la hora cuando se envió el mensaje.

La indicación de fecha y hora se establece mediante la llamadaIMessageProducer.send() cuando se envía el mensaje y se expresa en milisegundos desde las 00:00:00 GMT del 1 de enero de 1970.

Para un mensaje recién creado que no se ha enviado, la indicación de fecha y hora es 0, a menos que la aplicación emisora establezca una indicación de fecha y hora diferente estableciendo JMSTimestamp. Para un mensaje que se ha recibido, el método devuelve la indicación de fecha y hora que estableció la llamada IMessageProducer.send() cuando se envió el mensaje, a menos que la aplicación receptora cambie la indicación de fecha y hora estableciendo JMSTimestamp.

Excepciones:

- XMSException

Notas:

1. Si la indicación de fecha y hora no está definida, el método devuelve 0 pero no lanza ninguna excepción.

JMSType - Obtener y establecer JMSType

Interfaz:

```
String JMSType
{
  get;
  set;
}
```

Obtener y establecer el tipo de mensaje.

El valor de JMSType es una serie que encapsula el tipo del mensaje. Si es necesaria una conversión de datos, este valor es el tipo después de la conversión.

Excepciones:

- XMSException

PropertyNames - Obtener propiedades

Interfaz:

```
System.Collections.IEnumerator PropertyNames
{
  get;
}
```

Obtener una enumeración de las propiedades de nombres del mensaje.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Acknowledge	Acusar recibo de este mensaje y todos los mensajes recibidos previamente con acuse de recibo recibidos por la sesión.
ClearBody	Borrar el cuerpo del mensaje.
ClearProperties	Borrar las propiedades del mensaje.

Método	Descripción
PropertyExists	Comprobar si el mensaje tiene una prioridad con el nombre especificado.

Acknowledge - Acusar recibo

Interfaz:

```
void  Acknowledge();
```

Acusar recibo de este mensaje y todos los mensajes recibidos previamente con acuse de recibo recibidos por la sesión.

Una aplicación puede invocar este método si la modalidad de acuse de recibo de la sesión es AcknowledgeMode.ClientAcknowledge. Las llamadas al método no tienen ningún efecto si la sesión tiene cualquier otra modalidad de acuse de recibo o si ha realizado una transacción.

Los mensajes que se han recibido pero que no tienen acuse de recibo se podrían volver a entregar.

Si desea más información sobre cómo acusar recibo de mensajes, consulte “Acuse de recibo de mensaje” en la página 28.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- IllegalStateException

ClearBody - Borrar cuerpo

Interfaz:

```
void  ClearBody();
```

Borrar el cuerpo del mensaje. Los campos de cabecera y las propiedades de mensaje no se borran.

Si una aplicación borra un cuerpo de mensaje, el cuerpo se deja en el mismo estado que un cuerpo vacío en un mensaje recién creado. El estado de un cuerpo vacío en un mensaje recién creado depende del tipo de cuerpo del mensaje. Si desea más información, consulte “Cuerpo de un mensaje de Un XMS” en la página 77.

Una aplicación puede borrar un cuerpo del mensaje en cualquier momento, independientemente del estado en el que se encuentra el cuerpo. Si un cuerpo del mensaje es de solo lectura, la única forma en la que una aplicación puede escribir en el cuerpo es que la aplicación borre primero el cuerpo.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

ClearProperties - Borrar propiedades

Interfaz:

```
void  ClearProperties();
```

Borrar las propiedades del mensaje. Los campos de cabecera y el cuerpo del mensaje no se borran.

Si una aplicación borra las propiedades de un mensaje, las propiedades pasan a poderse leer y escribir.

Una aplicación puede borrar las propiedades de un mensaje en cualquier momento, independientemente del estado en que están las propiedades. Si las propiedades de un mensaje son de solo lectura, la única forma en la que las propiedades se pueden grabar es que la aplicación borre primero las propiedades.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

PropertyExists - Comprobar si existe la propiedad

Interfaz:

```
Boolean PropertyExists(String propertyName);
```

Comprobar si el mensaje tiene una prioridad con el nombre especificado.

Parámetros:

propertyName (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:

- `True`, si el mensaje tiene una propiedad con el nombre especificado.
- `False`, si el mensaje no tiene una propiedad con el nombre especificado.

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IMessageConsumer

Una aplicación utiliza un consumidor de mensaje para recibir mensajes enviados a un destino.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageConsumer
```

Para obtener una lista de las propiedades definidas de XMS de un objeto MessageConsumer, consulte “Propiedades de MessageConsumer” en la página 200.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
MessageListener	Obtener el escucha de mensajes que está registrado con el consumidor de mensajes y registrar un escucha de mensajes con el consumidor de mensajes.
MessageSelector	Obtener el selector de mensajes para el consumidor de mensajes.

MessageListener - Obtener y establecer escucha de mensajes

Interfaz:

```
MessageListener MessageListener
{
  get;
  set;
}
```

Obtener el escucha de mensajes que está registrado con el consumidor de mensajes y registrar un escucha de mensajes con el consumidor de mensajes.

Si no hay ningún escucha de mensajes registrado con el consumidor de mensajes, MessageListener es nulo. Si un escucha de mensajes ya está registrado con el consumidor de mensajes, puede cancelar el registro especificando en su lugar un valor nulo.

Si desea más información sobre cómo utilizar escuchas de mensajes, consulte "Escuchas de mensajes y excepción en .NET" en la página 50.

Excepciones:

- XMSException

MessageSelector - Obtener selector de mensajes

Interfaz:

```
String MessageSelector
{
  get;
}
```

Obtener el selector de mensajes para el consumidor de mensajes. el valor de retorno es un objeto String que encapsula la expresión de selector de mensajes. Si es necesaria la conversión de datos, este valor es la expresión de selector de mensajes después de la conversión. Si el consumidor de mensajes no tiene un selector de mensajes, el valor de MessageSelector es un objeto String nulo.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar el consumidor de mensajes.
Receive	Reciba el siguiente mensaje para el consumidor de mensajes. La llamada espera un mensaje de forma indefinida, o hasta que se cierra el consumidor de mensajes.

Método	Descripción
Receive	Reciba el siguiente mensaje para el consumidor de mensajes. La llamada solo espera un mensaje durante un periodo de tiempo especificado, o hasta que se cierra el consumidor de mensajes.
ReceiveNoWait	Recibir el siguiente mensaje para el consumidor de mensajes si uno está disponible de forma inmediata.

Close - Cerrar consumidor de mensajes

Interfaz:

```
void  Close();
```

Cerrar el consumidor de mensajes.

Si una aplicación intenta cerrar un consumidor de mensajes que ya está cerrado, la llamada se ignora.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

Recibir - Recibir

Interfaz:

```
IMessage Receive();
```

Reciba el siguiente mensaje para el consumidor de mensajes. La llamada espera un mensaje de forma indefinida, o hasta que se cierra el consumidor de mensajes.

Parámetros:
Ninguna

Devuelve:
Un puntero del objeto Message. Si el consumidor de mensajes se cierra cuando la llamada está esperando un mensaje, el método devuelve un puntero de un objeto Message nulo.

Excepciones:

- XMSException

Recibir - Recibir (con un intervalo de espera)

Interfaz:

```
IMessage Receive(Int64 delay);
```

Reciba el siguiente mensaje para el consumidor de mensajes. La llamada solo espera un mensaje durante un periodo de tiempo especificado, o hasta que se cierra el consumidor de mensajes.

Parámetros:

delay (entrada)
El tiempo, en milisegundos, que espera un mensaje la llamada. Si especifica un intervalo de espera de 0, la llamada espera un mensaje de forma indefinida.

Devuelve:
Un puntero del objeto Message. Si no llega ningún mensaje durante el intervalo de espera, o si el consumidor de mensajes se cierra mientras la llamada está esperando un mensaje, el método devuelve un puntero de un objeto Message nulo, pero no lanza ninguna excepción.

Excepciones:

- XMSException

ReceiveNoWait - Recibir sin espera

Interfaz:

```
IMessage ReceiveNoWait();
```

Recibir el siguiente mensaje para el consumidor de mensajes si uno está disponible de forma inmediata.

Parámetros:
Ninguna

Devuelve:
Un puntero de un objeto Message. Si no hay ningún mensaje disponible de forma inmediata, el método devuelve un puntero de un objeto Message nulo.

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

MessageEOFException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageEOFException
```

XMS lanza esta excepción si XMS encuentra el final de una corriente de mensaje de bytes cuando una aplicación lee el cuerpo de un mensaje de bytes.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

MessageFormatException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageFormatException
```

XMS lanza esta excepción si XMS encuentra un mensaje con un formato que no es válido.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

IMessageListener (delegado)

Jerarquía de herencia:
Ninguna

Una aplicación utiliza un escucha de mensajes para recibir mensajes de forma asíncrona.

Delegado

Resumen de métodos:

Método	Descripción
MessageListener	Entregue un mensaje de forma asíncrona al consumidor de mensajes.

MessageListener - Escucha de mensajes

Interfaz:

```
public delegate void MessageListener(IMessage msg);
```

Entregue un mensaje de forma asíncrona al consumidor de mensajes.

Los métodos que implementan este delegado se pueden registrar con la conexión.

Si desea más información sobre cómo utilizar escuchas de mensajes, consulte “Escuchas de mensajes y excepción en .NET” en la página 50.

Parámetros:

mesg (entrada)
El objeto Message.

Devuelve:
Void

MessageNotReadableException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageNotReadableException
```

XMS lanza esta excepción si una aplicación intenta leer el cuerpo de un mensaje que es de solo escritura.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

MessageNotWritableException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageNotWritableException
```

XMS lanza esta excepción si una aplicación intenta escribir en el cuerpo de un mensaje que es de solo lectura.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

IMessageProducer

Una aplicación utiliza un productor de mensajes para enviar mensajes a un destino.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageProducer
```

Si desea una lista de las propiedades definidas de XMS de un objeto MessageProducer, consulte “Propiedades de MessageProducer” en la página 200.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedades de .NET	Descripción
DeliveryMode	Obtener y establecer la modalidad de entrega predeterminada para mensajes enviados por el productor de mensajes.
Destination	Obtener el destino para el productor de mensajes.
DisableMsgID	Obtener una indicación sobre si una aplicación receptora requiere que se incluyan identificadores de mensaje en los mensajes enviados por el productor de mensajes, e indicar si una aplicación receptora requiere que se incluyan identificadores de mensaje en los mensajes enviados por el productor de mensajes.
DisableMsgTS	Obtener una indicación sobre si una aplicación receptora requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes, e indicar si una aplicación receptora requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes.
Priority	Obtener y establecer la prioridad predeterminada para los mensajes enviados por el productor de mensajes.
TimeToLive	Obtener y establecer el periodo de tiempo predeterminado que existe un mensaje antes de caducar.

DeliveryMode - Obtener y establecer modalidad de entrega predeterminada

Interfaz:

```
DeliveryMode DeliveryMode
{
  get;
  set;
}
```

Obtener y establecer la modalidad de entrega predeterminada para mensajes enviados por el productor de mensajes.

La modalidad de entrega predeterminada tiene uno de los valores siguientes:

`DeliveryMode.Persistent`

`DeliveryMode.NonPersistent`

Para una conexión en tiempo real con un intermediario, el valor debe ser `DeliveryMode.NonPersistent`.

El valor predeterminado es `DeliveryMode.Persistent`, excepto para una conexión en tiempo real con un intermediario para el cual el valor predeterminado es `DeliveryMode.NonPersistent`.

Excepciones:

- XMSException

Destination - Obtener destino

Interfaz:

```
IDestination Destination
{
  get;
}
```

Obtener el destino para el productor de mensajes.

Parámetros:
Ninguna

Devuelve:
El objeto Destination. Si el productor de mensajes no tiene un destino, el método devuelve un objeto Destination nulo.

Excepciones:

- XMSException

DisableMsgID - Obtener y establecer Inhabilitar distintivo de ID de mensaje

Interfaz:

```
Boolean DisableMessageID
{
  get;
  set;
}
```

Obtener una indicación sobre si una aplicación receptora requiere que se incluyan identificadores de mensaje en los mensajes enviados por el productor de mensajes, e indicar si una aplicación receptora requiere que se incluyan identificadores de mensaje en los mensajes enviados por el productor de mensajes.

En una conexión con un gestor de colas, o en una conexión en tiempo real con un intermediario, este distintivo se ignora. En una conexión con un bus de integración de servicios, se respeta el distintivo.

DisabledMsgID tiene los valores siguientes:

- `True`, si una aplicación receptora no requiere que se incluyan identificadores de mensaje en los mensajes enviados por el productor de mensajes.
- `False`, si una aplicación requiere que se incluyan identificadores de mensajes en los mensajes enviados por el productor de mensajes.

Excepciones:

- XMSException

DisableMsgTS - Obtener y establecer Inhabilitar distintivo de indicación de fecha y hora

Interfaz:

```
Boolean DisableMessageTimestamp
{
  get;
```

```
  set;
}
```

Obtener una indicación sobre si una aplicación receptora requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes, e indicar si una aplicación receptora requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes.

En una conexión de tiempo real con un intermediario, se ignora este distintivo. En una conexión con un gestor de colas, o en una conexión con un bus de integración de servicios, se respeta el distintivo.

DisableMsgTS tiene los valores siguientes:

- `True`, si una aplicación receptora no requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes.
- `False`, si una aplicación receptora requiere que se incluyan indicaciones de fecha y hora en los mensajes enviados por el productor de mensajes.

Devuelve:

Excepciones:

- XMSException

Priority - Obtener y establecer prioridad predeterminada

Interfaz:

```
Int32 Priority
{
  get;
  set;
}
```

Obtener y establecer la prioridad predeterminada para los mensajes enviados por el productor de mensajes.

El valor de la prioridad de mensaje predeterminada es un entero dentro del rango de `0`, la prioridad más baja, a 9, la prioridad más alta.

En una conexión en tiempo real con un intermediario, se ignora la prioridad de un mensaje.

Excepciones:

- XMSException

TimeToLive - Obtener y establecer tiempo de vida

Interfaz:

```
Int64 TimeToLive
{
  get;
  set;
}
```

Obtener y establecer el periodo de tiempo predeterminado que existe un mensaje antes de caducar.

El tiempo se mide a partir del momento cuando el productor de mensajes envía el mensaje y el tiempo de vida predeterminado en milisegundos. Un valor de 0 indica que un mensaje no caduca nunca.

Para una conexión en tiempo real con un intermediario, este valor siempre es 0.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar el productor de mensajes.
Send	Enviar un mensaje al destino que se ha especificado al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, la prioridad y el tiempo de vida predeterminados del productor de mensajes.
Send	Enviar un mensaje al destino que se ha especificado al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, prioridad y tiempo de vida especificados.
Send	Enviar un mensaje a un destino especificado si está utilizando un productor de mensajes para el que no se ha especificado ningún destino al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, la prioridad y el tiempo de vida predeterminados del productor de mensajes.
Send	Enviar un mensaje a un destino especificado si está utilizando un productor de mensajes para el que no se ha especificado ningún destino al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, prioridad y tiempo de vida especificados.

Close - Cerrar productor de mensajes

Interfaz:

```
void  Close();
```

Cerrar el productor de mensajes.

Si una aplicación intenta cerrar un productor de mensajes que ya está cerrado, se ignora la llamada.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

Send - Enviar

Interfaz:

```
void Send(IMessage msg) ;
```

Enviar un mensaje al destino que se ha especificado al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, la prioridad y el tiempo de vida predeterminados del productor de mensajes.

Parámetros:

msg (entrada)
El objeto Message.

Devuelve:
Void

Excepciones:

- XMSException

- MessageFormatException
- InvalidDestinationException

Send - Enviar (especificando una modalidad de entrega, prioridad y tiempo de vida)

Interfaz:

```
void Send(IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive);
```

Enviar un mensaje al destino que se ha especificado al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, prioridad y tiempo de vida especificados.

Parámetros:

msg (entrada)
El objeto Message.

deliveryMode (entrada)
La modalidad de entrega para el mensaje, que debe adoptar uno de los valores siguientes:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Para una conexión en tiempo real con un intermediario, el valor debe ser `DeliveryMode.NonPersistent`.

priority (entrada)
La prioridad del mensaje. El valor puede ser un entero dentro del rango de `0`, para la prioridad más baja, a 9, para la prioridad más alta. En una conexión en tiempo real con un intermediario, se ignora el valor.

timeToLive (entrada)
El tiempo de vida para el mensaje en milisegundos. Un valor de 0 significa que el mensaje no caduca nunca. Para una conexión en tiempo real con un intermediario, el valor debe ser 0.

Devuelve:
Void

Excepciones:

- XMSException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

Send - Enviar (a un destino especificado)

Interfaz:

```
void Send(IDestination dest, IMessage msg) ;
```

Enviar un mensaje a un destino especificado si está utilizando un productor de mensajes para el que no se ha especificado ningún destino al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, la prioridad y el tiempo de vida predeterminados del productor de mensajes.

Normalmente, puede especificar un destino cuando crea un productor de mensajes, pero si no lo hace, debe especificar un destino cada vez que envíe un mensaje.

Parámetros:

dest (entrada)
El objeto Destination.

msg (entrada)
El objeto Message.

Devuelve:
Void

Excepciones:

- XMSException
- MessageFormatException
- InvalidDestinationException

Send - Enviar (a un destino especificado, especificando una modalidad de entrega, prioridad y tiempo de vida)

Interfaz:

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

Enviar un mensaje a un destino especificado si está utilizando un productor de mensajes para el que no se ha especificado ningún destino al crear el productor de mensajes. Envíe el mensaje utilizando la modalidad de entrega, prioridad y tiempo de vida especificados.

Normalmente, puede especificar un destino cuando crea un productor de mensajes, pero si no lo hace, debe especificar un destino cada vez que envíe un mensaje.

Parámetros:

dest (entrada)
El objeto Destination.

msg (entrada)
El objeto Message.

deliveryMode (entrada)
La modalidad de entrega para el mensaje, que debe adoptar uno de los valores siguientes:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Para una conexión en tiempo real con un intermediario, el valor debe ser `DeliveryMode.NonPersistent`.

priority (entrada)
La prioridad del mensaje. El valor puede ser un entero dentro del rango de 0, para la prioridad más baja, a 9, para la prioridad más alta. En una conexión en tiempo real con un intermediario, se ignora el valor.

timeToLive (entrada)
El tiempo de vida para el mensaje en milisegundos. Un valor de 0 significa que el mensaje no caduca nunca. Para una conexión en tiempo real con un intermediario, el valor debe ser 0.

Devuelve:
Void

Excepciones:

- XMSException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IObjectMessage

Un mensaje de objeto es un mensaje cuyo cuerpo consta de un objeto Java o .NET serializado.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IObjectMessage
```

Referencia relacionada

Mensajes de objeto

El cuerpo de un mensaje de objeto contiene un objeto Java u objeto .NET serializado.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
Object	Obtener y establecer el objeto que forma el cuerpo del mensaje de objeto.

Object - Obtener y establecer objeto como bytes

Interfaz:

```
System.Object Object
    {
      get;
      set;
    }

    Byte[] GetObject();
```

Obtener y establecer el objeto que forma el cuerpo del mensaje de objeto.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException
- MessageNotWritableException

Propiedades y métodos heredados

Las propiedades siguientes se heredan de la interfaz IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Los métodos siguientes se heredan de la interfaz IMessage:

clearBody, clearProperties, PropertyExists

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IPropertyContext

IPropertyContext es una superclase abstracta que contiene métodos que obtienen y establecen propiedades. Estos métodos son heredados por otras clases.

Jerarquía de herencia:

Ninguna

Métodos

Resumen de métodos:

Método	Descripción
GetBooleanProperty	Obtener el valor de la propiedad booleana con el nombre especificado.
GetByteProperty	Obtener el valor de la propiedad de byte identificada por el nombre.
GetBytesProperty	Obtener el valor de la propiedad de matriz de bytes identificada por el nombre.
GetCharProperty	Obtener el valor de la propiedad de carácter de 2-bytes identificada por el nombre.
GetDoubleProperty	Obtener el valor de la propiedad de coma flotante de precisión doble identificada por el nombre.
GetFloatProperty	Obtener el valor de la propiedad de coma flotante identificada por el nombre.
GetIntProperty	Obtener el valor de la propiedad de entero identificada por el nombre.
GetLongProperty	Obtener el valor de la propiedad de entero largo identificada por el nombre.
GetObjectProperty	Obtener el valor y el tipo de datos de la propiedad identificada por el nombre.
GetShortProperty	Obtener el valor de la propiedad de entero corto identificada por el nombre.
GetStringProperty	Obtener el valor de la propiedad de serie identificada por el nombre.
SetBooleanProperty	Establecer el valor de la propiedad booleana identificada por el nombre.
SetByteProperty	Establecer el valor de la propiedad de byte identificada por el nombre.
SetBytesProperty	Establecer el valor de la propiedad de matriz de bytes identificada por el nombre.
SetCharProperty	Establecer el valor de la propiedad de carácter de 2-bytes identificada por el nombre.
SetDoubleProperty	Establecer el valor de la propiedad de coma flotante de precisión doble identificada por el nombre.
SetFloatProperty	Establecer el valor de la propiedad de coma flotante identificada por el nombre.
SetIntProperty	Establecer el valor de la propiedad de entero identificada por el nombre.
SetLongProperty	Establecer el valor de la propiedad de entero largo identificada por el nombre.

Método	Descripción
SetObjectProperty	Establecer el valor y el tipo de datos de una propiedad identificada por el nombre.
SetShortProperty	Establecer el valor de la propiedad de entero corto identificada por el nombre.
SetStringProperty	Establecer el valor de la propiedad de serie identificada por el nombre.

GetBooleanProperty - Obtener propiedad booleana

Interfaz:

```
Boolean GetBooleanProperty(String property_name);
```

Obtener el valor de la propiedad booleana con el nombre especificado.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetByteProperty - Obtener propiedad de byte

Interfaz:

```
Byte    GetByteProperty(String property_name) ;
Int16   GetSignedByteProperty(String property_name) ;
```

Obtener el valor de la propiedad de byte identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetBytesProperty - Obtener propiedad de matriz de bytes

Interfaz:

```
Byte[]  GetBytesProperty(String property_name) ;
```

Obtener el valor de la propiedad de matriz de bytes identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El número de bytes de la matriz.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetCharProperty - Obtener propiedad de carácter

Interfaz:

```
Char    GetCharProperty(String property_name) ;
```

Obtener el valor de la propiedad de carácter de 2-bytes identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetDoubleProperty - Obtener propiedad de coma flotante de precisión doble

Interfaz:

```
Double  GetDoubleProperty(String property_name) ;
```

Obtener el valor de la propiedad de coma flotante de precisión doble identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetFloatProperty - Obtener propiedad de coma flotante

Interfaz:

```
Single  GetFloatProperty(String property_name) ;
```

Obtener el valor de la propiedad de coma flotante identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetIntProperty - GetIntProperty

Interfaz:

```
Int32   GetIntProperty(String property_name) ;
```

Obtener el valor de la propiedad de entero identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetLongProperty - Obtener propiedad de entero largo

Interfaz:

```
Int64   GetLongProperty(String property_name) ;
```

Obtener el valor de la propiedad de entero largo identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetObjectProperty - Obtener propiedad de objeto

Interfaz:

```
Object  GetObjectProperty( String property_name) ;
```

Obtener el valor y el tipo de datos de la propiedad identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad, que adopta uno de los tipos de objeto siguientes:

```
Boolean
Byte
```

```
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetShortProperty - Obtener propiedad de entero corto

Interfaz:

```
Int16   GetShortProperty(String property_name) ;
```

Obtener el valor de la propiedad de entero corto identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
El valor de la propiedad.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

GetStringProperty - GetStringProperty

Interfaz:

```
String  GetStringProperty(String property_name) ;
```

Obtener el valor de la propiedad de serie identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

Devuelve:
Un objeto String que encapsula la serie que es el valor de la propiedad. Si es necesaria una conversión de datos, este valor es la serie después de la conversión.

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

SetBooleanProperty - Establecer propiedad booleana

Interfaz:

```
void SetBooleanProperty( String property_name,  Boolean value) ;
```

Establecer el valor de la propiedad booleana identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetByteProperty - Establecer propiedad de byte

Interfaz:

```
void SetByteProperty( String property_name,  Byte value) ;
      void SetSignedByteProperty( String property_name,  Int16 value) ;
```

Establecer el valor de la propiedad de byte identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetBytesProperty - Establecer propiedad de matriz de bytes

Interfaz:

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

Establecer el valor de la propiedad de matriz de bytes identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad, que es una matriz de bytes.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetCharProperty - Establecer propiedad de carácter

Interfaz:

```
void SetCharProperty( String property_name,  Char value) ;
```

Establecer el valor de la propiedad de carácter de 2-bytes identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetDoubleProperty - Establecer propiedad de coma flotante de precisión doble

Interfaz:

```
void SetDoubleProperty( String property_name,  Double value) ;
```

Establecer el valor de la propiedad de coma flotante de precisión doble identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetFloatProperty - Establecer propiedad de coma flotante

Interfaz:

```
void SetFloatProperty( String property_name,  Single value) ;
```

Establecer el valor de la propiedad de coma flotante identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetIntProperty - Establecer propiedad de entero

Interfaz:

```
void SetIntProperty( String property_name,  Int32 value) ;
```

Establecer el valor de la propiedad de entero identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetLongProperty - Establecer propiedad de entero largo

Interfaz:

```
void SetLongProperty( String property_name,  Int64 value) ;
```

Establecer el valor de la propiedad de entero largo identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException

- MessageNotWritableException

SetObjectProperty - Establecer propiedad de objeto

Interfaz:

```
void SetObjectProperty( String property_name,  Object value) ;
```

Establecer el valor y el tipo de datos de una propiedad identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

objectType (entrada)
El valor de la propiedad, que debe adoptar uno de los tipos de objeto siguientes:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

value (entrada)
El valor de la propiedad como una matriz de bytes.

length (entrada)
El número de bytes de la matriz.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetShortProperty - Establecer propiedad de entero corto

Interfaz:

```
void SetShortProperty( String property_name,  Int16 value) ;
```

Establecer el valor de la propiedad de entero corto identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
El valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

SetStringProperty - Establecer propiedad de serie

Interfaz:

```
void SetStringProperty( String property_name,  String value);
```

Establecer el valor de la propiedad de serie identificada por el nombre.

Parámetros:

property_name (entrada)
Un objeto String que encapsula el nombre de la propiedad.

value (entrada)
Un objeto String que encapsula la serie que es el valor de la propiedad.

Devuelve:
Void

Contexto de hebras
Se determina mediante la subclase.

Excepciones:

- XMSException
- MessageNotWritableException

IQueueBrowser

Una aplicación utiliza un examinador de colas para examinar mensajes en una cola sin eliminarlas.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
   |
   +----IBM.XMS.IQueueBrowser
```

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
MessageSelector	Obtener el selector de mensajes para el examinador de colas.
Queue	Obtener la cola asociada al examinador de colas como un objeto de destino que representa la cola.

MessageSelector - Obtener selector de mensajes

Interfaz:

```
String MessageSelector
{
  get;
}
```

Obtener el selector de mensajes para el examinador de colas.

El selector de mensajes es un objeto String que encapsula la expresión de selector de mensajes. Si es necesaria la conversión de datos, este valor es la expresión de selector de mensajes después de la

conversión. Si el examinador de colas no tiene un selector de mensajes, el método devuelve un objeto String nulo.

Excepciones:

- XMSException

Queue - Obtener cola

Interfaz:

```
IDestination Queue
{
  get;
}
```

Obtener la cola asociada al examinador de colas como un objeto de destino que representa la cola.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar el examinador de colas.
GetEnumerator	Obtener una lista de los mensajes de la cola.

Close - Cerrar examinador de colas

Interfaz:

```
void  Close();
```

Cerrar el examinador de colas.

Si una aplicación intenta cerrar un examinador de colas que está cerrado, la llamada se ignora.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException

GetEnumerator - Obtener mensajes

Interfaz:

```
IEnumerator GetEnumerator();
```

Obtener una lista de los mensajes de la cola.

El método devuelve un enumerador que encapsula una lista de objetos Message. El orden de los objetos Message es el mismo que el orden en el cual se recuperarán los mensajes de la cola. La aplicación puede utilizar el enumerador para examinar cada mensaje a su vez.

El enumerador se actualiza de forma dinámica a medida que los mensajes se colocan en la cola y se eliminan de la cola. Cada vez que la aplicación llama a IEnumerator.MoveNext() para examinar el siguiente mensaje en la cola, el mensaje refleja el contenido actual de la cola.

Si una aplicación llama a este método más de una vez para un examinador de colas, cada llamada devuelve un nuevo enumerador. Por lo tanto, la aplicación puede utilizar más de un enumerador para examinar los mensajes de una cola y mantener varias posiciones dentro de la cola.

Parámetros:
Ninguna

Devuelve:
El objeto Iterator.

Excepciones:

- XMSException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

Solicitante

Una aplicación utiliza un solicitante para enviar un mensaje de solicitud y, después, esperar una respuesta y recibirla.

Jerarquía de herencia:
Ninguna

Constructores

Resumen de constructores

Constructor	Descripción
Solicitante	Crear un solicitante.

Requestor - Crear solicitante

Interfaz:

```
Requestor(ISession sess, IDestination dest);
```

Crear un solicitante.

Parámetros:

sess (entrada)
Un objeto Session. La sesión no debe haber realizado una transacción y debe tener una de las modalidades de acuse de recibo siguientes:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

dest (entrada)
Un objeto Destination que representa el destino adonde la aplicación puede enviar mensajes de solicitud.

Contexto de hebras
La sesión asociada al solicitante

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar el solicitante.
Solicitar	Enviar un mensaje de solicitud y, después, esperar una respuesta y recibirla de la aplicación que recibe el mensaje de solicitud.

Close - Cerrar solicitante

Interfaz:

```
void  Close();
```

Cerrar el solicitante.

Si una aplicación intenta cerrar un solicitante que ya está cerrado, se ignora la llamada.

Nota: Cuando una aplicación cierra un solicitante, la sesión asociada no se cierra. En este sentido, XMS se comporta de forma diferente en comparación con JMS.

Parámetros:
Ninguna

Devuelve:
Void

Contexto de hebras
Cualquiera

Excepciones:

- XMSException

Solicitar - Solicitar respuesta

Interfaz:

```
IMessage Request(IMessage requestMessage);
```

Enviar un mensaje de solicitud y, después, esperar una respuesta y recibirla de la aplicación que recibe el mensaje de solicitud.

Se bloquea una llamada a este método hasta que se recibe una respuesta o hasta que finaliza la sesión, lo que se produzca antes.

Parámetros:

requestMessage (entrada)
El objeto Message que encapsula el mensaje de solicitud.

Devuelve:
Un puntero del objeto Message que encapsula el mensaje de respuesta.

Contexto de hebras
La sesión asociada al solicitante

Excepciones:

- XMSException

ResourceAllocationException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
```

```
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.ResourceAllocationException
```

XMS lanza esta excepción si XMS no puede asignar los recursos necesarios para un método.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

SecurityException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.SecurityException
```

XMS lanza esta excepción si se rechazan el identificador de usuario y la contraseña proporcionados para autenticar una aplicación. XMS también lanza esta excepción si una comprobación de autoridad falla e impide que se complete un método.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

ISession

Una sesión es un contexto de hebra única para enviar y recibir mensajes.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.ISession
```

Si desea una lista de propiedades definidas de XMS de un objeto Session, consulte “Propiedades de sesión” en la página 200.

Propiedades .NET

Resumen de propiedades .NET

Propiedad .NET	Descripción
AcknowledgeMode	Se obtiene la modalidad de acuse de recibo de la sesión.
Transacted	Determinar si la sesión es una sesión con transacción.

AcknowledgeMode - Obtener modalidad de acuse de recibo

Interfaz:

```
AcknowledgeMode AcknowledgeMode
    {
      get;
    }
```

Se obtiene la modalidad de acuse de recibo de la sesión.

La modalidad de acuse de recibo se especifica cuando se crea la sesión.

Siempre que la sesión no haya realizado una transacción, la modalidad de acuse de recibo tiene uno de los valores siguientes:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Si desea más información sobre las modalidades de acuse de recibo, consulte "Acuse de recibo de mensaje" en la página 28.

Una sesión que ha realizado una transacción no tiene modalidad de acuse de recibo. Si la sesión es una sesión con transacción, en su lugar el método devuelve `AcknowledgeMode.SessionTransacted`.

Excepciones:

- XMSException

Transacted - Determinar si es transaccional

Interfaz:

```
Boolean Transacted
    {
      get;
    }
```

Determinar si la sesión es una sesión con transacción.

El estado con transacción es:

- `True`, si la sesión es una sesión con transacción.
- `False`, si la sesión no es una sesión con transacción.

Para una conexión en tiempo real con un intermediario, el método siempre devuelve `False`.

Excepciones:

- XMSException

Métodos

Resumen de métodos:

Método	Descripción
Cerrar	Cerrar la sesión.
Commit	Confirmar todos los mensajes procesados en la transacción actual.
CreateBrowser	Crear un examinador de colas para la cola especificada.
CreateBrowser	Crear un examinador de colas para la cola especificada utilizando un selector de mensajes.
CreateBytesMessage	Crear un mensaje de bytes.
CreateConsumer	Crear un consumidor de mensajes para el destino especificado.
CreateConsumer	Crear un consumidor de mensajes para el destino especificado utilizando un selector de mensajes.
CreateConsumer	Crear un consumidor de mensajes para el destino especificado utilizando un selector de mensajes y, si el destino es un tema, especificando si el consumidor de mensajes recibe los mensajes publicados por su propia conexión.
CreateDurableSubscriber	Crear un suscriptor duradero para el tema especificado.

Método	Descripción
CreateDurableSubscriber	Crear un suscriptor duradero para el tema especificado utilizando un selector de mensajes y especificando si el suscriptor duradero recibe los mensajes publicados por su propia conexión.
CreateMapMessage	Crear un mensaje de correlación.
CreateMessage	Crear un mensaje que no tiene un cuerpo.
CreateObjectMessage	Crear un mensaje de objeto.
CreateProducer	Crear un productor de mensajes para enviar mensajes al destino especificado.
CreateQueue	Crear un objeto Destination para representar una cola en el servidor de mensajería.
CreateStreamMessage	Crear un mensaje de corriente de datos.
CreateTemporaryQueue	Crear una cola temporal.
CreateTemporaryTopic	Crear un tema temporal.
CreateTextMessage	Crear un mensaje de texto con un cuerpo vacío.
CreateTextMessage	Crear un mensaje de texto cuyo cuerpo se ha inicializado con el texto especificado.
CreateTopic	Crear un objeto Destination para representar un tema.
Recover	Recuperar la sesión.
Rollback	Se retrotraen todos los mensajes procesados en la transacción actual.
Unsubscribe	Suprimir una suscripción duradera.

Close - Cerrar sesión

Interfaz:

```
void Close();
```

Cerrar la sesión. Si la sesión es una sesión con transacción, se retrotrae cualquier transacción en curso.

Si una aplicación intenta cerrar una sesión que ya está cerrada, la llamada se ignora.

Parámetros:
Ninguna

Devuelve:
Void

Contexto de hebras
Cualquiera

Excepciones:

- XMSException

Commit - Confirmar

Interfaz:

```
void Commit();
```

Confirmar todos los mensajes procesados en la transacción actual.

La sesión debe ser una sesión con transacción.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- IllegalStateException
- TransactionRolledBackException

CreateBrowser - Crear examinador de colas

Interfaz:

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

Crear un examinador de colas para la cola especificada.

Parámetros:

queue (entrada)
Un objeto Destination que representa la cola.

Devuelve:
El objeto QueueBrowser.

Excepciones:

- XMSException
- InvalidDestinationException

CreateBrowser - Crear examinador de colas (con selector de mensajes)

Interfaz:

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

Crear un examinador de colas para la cola especificada utilizando un selector de mensajes.

Parámetros:

queue (entrada)
Un objeto Destination que representa la cola.

selector (entrada)
Un objeto String que encapsula una expresión de selector de mensajes. Solo se entregan al examinador de colas los mensajes con propiedades que coinciden con la expresión de selector de mensajes.

Un objeto String nulo significa que no hay ningún selector de mensajes para el examinador de colas.

Devuelve:
El objeto QueueBrowser.

Excepciones:

- XMSException
- InvalidDestinationException
- InvalidSelectorException

CreateBytesMessage - Crear mensaje de bytes

Interfaz:

```
IBytesMessage CreateBytesMessage();
```

Crear un mensaje de bytes.

Parámetros:
Ninguna

Devuelve:
El objeto BytesMessage.

Excepciones:

- XMSException
- IllegalStateException (La sesión está cerrada)

CreateConsumer - Crear consumidor

Interfaz:

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

Crear un consumidor de mensajes para el destino especificado.

Parámetros:

dest (entrada)
El objeto Destination.

Devuelve:
El objeto MessageConsumer.

Excepciones:

- XMSException
- InvalidDestinationException

CreateConsumer - Crear consumidor (con selector de mensajes)

Interfaz:

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector) ;
```

Crear un consumidor de mensajes para el destino especificado utilizando un selector de mensajes.

Parámetros:

dest (entrada)
El objeto Destination.

selector (entrada)
Un objeto String que encapsula una expresión de selector de mensajes. Solo se entregan al consumidor de mensajes los mensajes con propiedades que coinciden con la expresión de selector de mensajes.

Un objeto String nulo significa que no hay ningún selector de mensajes para el consumidor de mensajes.

Devuelve:
El objeto MessageConsumer.

Excepciones:

- XMSException
- InvalidDestinationException

- InvalidSelectorException

CreateConsumer - Crear consumidor (con el selector de mensajes y el distintivo de mensaje local)

Interfaz:

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector,
                                Boolean noLocal) ;
```

Crear un consumidor de mensajes para el destino especificado utilizando un selector de mensajes y, si el destino es un tema, especificando si el consumidor de mensajes recibe los mensajes publicados por su propia conexión.

Parámetros:

dest (entrada)
El objeto Destination.

selector (entrada)
Un objeto String que encapsula una expresión de selector de mensajes. Solo se entregan al consumidor de mensajes los mensajes con propiedades que coinciden con la expresión de selector de mensajes.

Un objeto String nulo significa que no hay ningún selector de mensajes para el consumidor de mensajes.

noLocal (entrada)
El valor `True` significa que el consumidor de mensajes no recibe los mensajes publicados por su propia conexión. El valor `False` significa que el consumidor de mensajes no recibe los mensajes publicados por su propia conexión. El valor predeterminado es `False`.

Devuelve:
El objeto MessageConsumer.

Excepciones:

- XMSException
- InvalidDestinationException
- InvalidSelectorException

CreateDurableSubscriber - Crear suscriptor duradero

Interfaz:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription) ;
```

Crear un suscriptor duradero para el tema especificado.

Este método no es válido para una conexión en tiempo real con un intermediario.

Si desea más información sobre suscriptores duraderos, consulte “Suscriptores duraderos” en la página 36.

Parámetros:

dest (entrada)
Un objeto Destination que representa el tema. El tema no debe ser un tema temporal.

subscription (entrada)
Un objeto String encapsula un nombre que identifica la suscripción duradera. El nombre debe ser exclusivo dentro del identificador de cliente para la conexión.

Devuelve:
El objeto MessageConsumer que representa el suscriptor duradero.

Excepciones:

- XMSException
- InvalidDestinationException

CreateDurableSubscriber - Crear suscriptores duraderos (con selector de mensajes y distintivo de mensaje local)

Interfaz:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription,
                                         String selector,
                                         Boolean noLocal) ;
```

Crear un suscriptor duradero para el tema especificado utilizando un selector de mensajes y especificando si el suscriptor duradero recibe los mensajes publicados por su propia conexión.

Este método no es válido para una conexión en tiempo real con un intermediario.

Si desea más información sobre suscriptores duraderos, consulte "Suscriptores duraderos" en la página 36.

Parámetros:

dest (entrada)
Un objeto Destination que representa el tema. El tema no debe ser un tema temporal.

subscription (entrada)
Un objeto String encapsula un nombre que identifica la suscripción duradera. El nombre debe ser exclusivo dentro del identificador de cliente para la conexión.

selector (entrada)
Un objeto String que encapsula una expresión de selector de mensajes. Solo se devuelven al suscriptor duradero los mensajes con propiedades que coinciden con la expresión de selector de mensajes.

Un objeto String nulo significa que no hay ningún selector de mensajes para el suscriptor duradero.

noLocal (entrada)
El valor `True` significa que el suscriptor duradero no recibe los mensajes publicados por su propia conexión. El valor `False` significa que el suscriptor duradero no recibe los mensajes publicados por su propia conexión. El valor predeterminado es `False`.

Devuelve:
El objeto MessageConsumer que representa el suscriptor duradero.

Excepciones:

- XMSException
- InvalidDestinationException
- InvalidSelectorException

CreateMapMessage - Crear mensaje de correlación

Interfaz:

```
IMapMessage CreateMapMessage();
```

Crear un mensaje de correlación.

Parámetros:
Ninguna

Devuelve:
El objeto MapMessage.

Excepciones:

- XMSException
- IllegalStateException (La sesión está cerrada)

CreateMessage - Crear mensaje

Interfaz:

```
IMessage CreateMessage();
```

Crear un mensaje que no tiene un cuerpo.

Parámetros:
Ninguna

Devuelve:
El objeto Message.

Excepciones:

- XMSException
- IllegalStateException (La sesión está cerrada)

CreateObjectMessage - Crear mensaje de objeto

Interfaz:

```
IObjectMessage CreateObjectMessage();
```

Crear un mensaje de objeto.

Parámetros:
Ninguna

Devuelve:
El objeto ObjectMessage.

Excepciones:

- XMSException
- IllegalStateException (La sesión está cerrada)

CreateProducer - Crear productor

Interfaz:

```
IMessageProducer CreateProducer(IDestination dest) ;
```

Crear un productor de mensajes para enviar mensajes al destino especificado.

Parámetros:

dest (entrada)
El objeto Destination.

Si especifica un objeto Destination nulo, el productor de mensajes se crea sin un destino. En este caso, la aplicación debe especificar un destino cada vez que utiliza el productor de mensajes envía un mensaje.

Devuelve:
El objeto MessageProducer.

Excepciones:

- XMSException
- InvalidDestinationException

CreateQueue - Crear cola

Interfaz:

```
IDestination CreateQueue(String queue) ;
```

Crear un objeto Destination para representar una cola en el servidor de mensajería.

Este método no crea la cola en el servidor de mensajería. Debe crear la cola antes de que una aplicación pueda llamar a este método.

Parámetros:

queue (entrada)
Un objeto String que encapsula el nombre de la cola, o encapsula un identificador uniforme de recursos (URI) que identifica la cola.

Devuelve:
El objeto Destination que representa la cola.

Excepciones:

- XMSException

CreateStreamMessage - Crear mensaje de corriente de datos

Interfaz:

```
IStreamMessage CreateStreamMessage();
```

Crear un mensaje de corriente de datos.

Parámetros:
Ninguna

Devuelve:
El objeto StreamMessage.

Excepciones:

- XMSException
- XMS_ILLEGAL_STATE_EXCEPTION

CreateTemporaryQueue - Crear cola temporal

Interfaz:

```
IDestination CreateTemporaryQueue() ;
```

Crear una cola temporal.

El ámbito de la cola temporal es la conexión. Solo las sesiones creadas por la conexión pueden utilizar la cola temporal.

La cola temporal permanece hasta que se suprime de forma explícita, o finaliza la conexión, lo que suceda antes.

Si desea más información sobre colas temporales, consulte “Destinos temporales” en la página 34.

Parámetros:
Ninguna

Devuelve:
El objeto Destination que representa la cola temporal.

Excepciones:

- XMSException

CreateTemporaryTopic - Crear tema temporal

Interfaz:

```
IDestination CreateTemporaryTopic() ;
```

Crear un tema temporal.

El ámbito del tema temporal es la conexión. Solo las sesiones creadas por la conexión pueden utilizar el tema temporal.

El tema temporal permanece hasta que se suprime de forma explícita, o finaliza la conexión, lo que suceda antes.

Si desea más información sobre los temas temporales, consulte "Destinos temporales" en la página 34.

Parámetros:
Ninguna

Devuelve:
El objeto Destination que representa el tema temporal.

Excepciones:

- XMSException

CreateTextMessage - Crear mensaje de texto

Interfaz:

```
ITextMessage CreateTextMessage();
```

Crear un mensaje de texto con un cuerpo vacío.

Parámetros:
Ninguna

Devuelve:
El objeto TextMessage.

Excepciones:

- XMSException

CreateTextMessage - Crear mensaje de texto (inicializado)

Interfaz:

```
ITextMessage CreateTextMessage(String initialValue);
```

Crear un mensaje de texto cuyo cuerpo se ha inicializado con el texto especificado.

Parámetros:

initialValue (entrada)
Un objeto String que encapsula el texto para inicializar el cuerpo del mensaje de texto.

Ninguna

Devuelve:
El objeto TextMessage.

Excepciones:

- XMSException

CreateTopic - Crear tema

Interfaz:

```
IDestination CreateTopic(String topic) ;
```

Crear un objeto Destination para representar un tema.

Parámetros:

topic (entrada)
Un objeto String que encapsula el nombre del tema, o encapsula un identificador uniforme de recursos (URI) que identifica el tema.

Devuelve:
El objeto Destination que representa el tema.

Excepciones:

- XMSException

Recover - Recuperar

Interfaz:

```
void Recover();
```

Recuperar la sesión. Se detiene la entrega de mensaje y, después, se reinicia con el mensaje sin acuse de recibo más antiguo.

La sesión no debe ser una sesión con transacción.

Si desea más información sobre cómo recuperar una sesión, consulte "Acuse de recibo de mensaje" en la página 28.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- IllegalStateException

Rollback - Retrotraer

Interfaz:

```
void Rollback();
```

Se retrotraen todos los mensajes procesados en la transacción actual.

La sesión debe ser una sesión con transacción.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- IllegalStateException

Unsubscribe - Anular suscripción

Interfaz:

```
void Unsubscribe(String subscription);
```

Suprimir una suscripción duradera. El servidor de mensajería suprime el registro de la suscripción duradera que está manteniendo y no envía ningún mensaje más al suscriptor duradero.

Una aplicación no puede suprimir una suscripción duradera bajo ninguna de las circunstancias siguientes:

- Mientras haya un consumidor de mensajes activo para la suscripción duradera
- Mientras un mensaje consumido forme parte de una transacción pendiente
- Mientras un mensaje consumido no tenga acuse de recibo

Este método no es válido para una conexión en tiempo real con un intermediario.

Parámetros:

subscription (entrada)
Un objeto String que encapsula el nombre que identifica la suscripción duradera.

Devuelve:
Void

Excepciones:

- XMSException
- InvalidDestinationException
- IllegalStateException

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IStreamMessage

Un mensaje de corriente de datos es un mensaje cuyo cuerpo está formado por una corriente de valores, donde cada valor tiene un tipo de datos asociado. El contenido del cuerpo se escribe y se lee de forma secuencial.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IStreamMessage
```

Cuando una aplicación lee un valor de la corriente de datos de mensaje, el valor se puede convertir mediante XMS a otro tipo de datos. Si desea más información sobre esta forma de conversión implícita, consulte “Mensajes de corriente de datos” en la página 81.

Referencia relacionada

Mensajes de corriente de datos
El cuerpo de un mensaje de corriente de datos contiene una secuencia de valores, donde cada valor tiene un tipo de datos asociado.

Métodos

Resumen de métodos:

Método	Descripción
ReadBoolean	Leer un valor booleano en la corriente de datos de mensaje.
ReadByte	Leer un entero de 8-bits firmado en la corriente de datos de mensaje.
ReadBytes	Leer una matriz de bytes en la corriente de datos de mensaje.

Método	Descripción
ReadChar	Leer un carácter de 2-bytes en la corriente de datos de mensaje.
ReadDouble	Leer un número de coma flotante de precisión doble de 8-bytes en la corriente de datos de mensaje.
ReadFloat	Leer un número de coma flotante de 4-bytes en la corriente de datos de mensaje.
ReadInt	Leer un entero de 32-bits firmado en la corriente de datos de mensaje.
ReadLong	Leer un entero de 64-bits firmado en la corriente de datos de mensaje.
ReadObject	Leer un valor en la corriente de datos de mensaje y devolver su tipo de datos.
ReadShort	Leer un entero de 16-bits firmado en la corriente de datos de mensaje.
ReadString	Leer una serie en la corriente de datos de mensaje.
Reset	Colocar el cuerpo del mensaje en la modalidad de solo lectura y volver a colocar el cursor al principio de la corriente de datos de mensaje.
WriteBoolean	Escribir un valor booleano en la corriente de datos de mensaje.
WriteByte	Escribir un byte en la corriente de datos de mensaje.
WriteBytes	Escribir una matriz de bytes en la corriente de datos de mensaje.
WriteChar	Escribir un carácter en la corriente de datos de mensaje como 2 bytes, primero el byte de orden superior.
WriteDouble	Convertir un número de coma flotante de precisión doble a un entero largo y escribir el entero largo en la corriente de datos de mensaje como 8 bytes, primero el byte de orden superior.
WriteFloat	Convertir una número de coma flotante a un entero y escribir el entero en la corriente de datos de mensaje como 4 bytes, primero el byte de orden superior.
WriteInt	Escribir un entero en la corriente de datos de mensaje como 4 bytes, primero el byte de orden superior.
WriteLong	Escribir un entero largo en la corriente de datos de mensaje como 8 bytes, primero el byte de orden superior.
WriteObject	Escribir un valor, con un tipo de datos especificado, en la corriente de datos de mensaje.
WriteShort	Escribir un entero corto en la corriente de datos de mensaje como 2 bytes, primero el byte de orden superior.
WriteString	Escribir serie en la corriente de datos de mensaje.

ReadBoolean - Leer valor booleano

Interfaz:

```
Boolean ReadBoolean();
```

Leer un valor booleano en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El valor booleano que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte - Leer byte

Interfaz:

```
Int16   ReadSignedByte();
    Byte    ReadByte();
```

Leer un entero de 8-bits firmado en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El byte que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes - Leer bytes

Interfaz:

```
Int32   ReadBytes(Byte[] array);
```

Leer una matriz de bytes en la corriente de datos de mensaje.

Parámetros:

array (entrada)
El almacenamiento intermedio que contiene la matriz de bytes que se ha leído y la longitud del almacenamiento intermedio en bytes.

Si el número de bytes de la matriz es menor o igual que la longitud del almacenamiento intermedio, se lee toda la matriz en el almacenamiento intermedio. Si el número de bytes de la matriz es mayor que la longitud del almacenamiento intermedio, el almacenamiento intermedio se rellena con parte de la matriz, y un cursor interno marca la posición del siguiente byte que se va a leer. Una llamada posterior a readBytes() lee bytes de la matriz empezando por la posición actual del cursor.

Si especifica un puntero nulo en la entrada, la llamada se salta la matriz de bytes sin leerla.

Devuelve:
El número de bytes que se van a leer en el almacenamiento intermedio. Si el almacenamiento intermedio se rellena parcialmente, el valor es menor que la longitud del almacenamiento intermedio, lo que indica que no quedan más bytes en la matriz para leer. Si no queda ningún byte para leer en la matriz antes de la llamada, el valor es XMSC_END_OF_BYTEARRAY.

Si especifica un puntero nulo en la entrada, el método no devuelve ningún valor.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadChar - Leer carácter

Interfaz:

```
Char    ReadChar();
```

Leer un carácter de 2-bytes en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El carácter que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble - Leer número de coma flotante de precisión doble

Interfaz:

```
Double  ReadDouble();
```

Leer un número de coma flotante de precisión doble de 8-bytes en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El número de coma flotante de precisión doble que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat - Leer número de coma flotante

Interfaz:

```
Single  ReadFloat();
```

Leer un número de coma flotante de 4-bytes en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El número de coma flotante que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt - Leer entero

Interfaz:

```
Int32   ReadInt();
```

Leer un entero de 32-bits firmado en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El entero que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong - Leer entero largo

Interfaz:

```
Int64   ReadLong();
```

Leer un entero de 64-bits firmado en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El entero largo que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadObject - Leer objeto

Interfaz:

```
Object  ReadObject();
```

Leer un valor en la corriente de datos de mensaje y devolver su tipo de datos.

Parámetros:
Ninguna

Devuelve:
El valor, que es uno de los tipos de objeto siguientes:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

Excepciones:
XMSException

ReadShort - Leer entero corto

Interfaz:

```
Int16   ReadShort();
```

Leer un entero de 16-bits firmado en la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
El entero corto que se ha leído.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

ReadString - Leer serie

Interfaz:

```
String  ReadString();
```

Leer una serie en la corriente de datos de mensaje. Si es necesario, XMS convierte los caracteres de la serie en la página de códigos local.

Parámetros:
Ninguna

Devuelve:
Un objeto String que encapsula la serie que se ha leído. Si es necesaria la conversión de datos, esta es la serie después de la conversión.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

Reset - Restablecer

Interfaz:

```
void  Reset();
```

Colocar el cuerpo del mensaje en la modalidad de solo lectura y volver a colocar el cursor al principio de la corriente de datos de mensaje.

Parámetros:
Ninguna

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotReadableException
- MessageEOFException

WriteBoolean - Escribir valor booleano

Interfaz:

```
void  WriteBoolean(Boolean value);
```

Escribir un valor booleano en la corriente de datos de mensaje.

Parámetros:

value (entrada)
El valor booleano que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteByte - Escribir byte

Interfaz:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Escribir un byte en la corriente de datos de mensaje.

Parámetros:

value (entrada)
El byte que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteBytes - Escribir bytes

Interfaz:

```
void  WriteBytes(Byte[] value);
```

Escribir una matriz de bytes en la corriente de datos de mensaje.

Parámetros:

value (entrada)
La matriz de bytes que se va a escribir.

length (entrada)
El número de bytes de la matriz.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteChar - Escribir carácter

Interfaz:

```
void  WriteChar(Char value);
```

Escribir un carácter en la corriente de datos de mensaje como 2 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El carácter que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteDouble - Escribir número de coma flotante de precisión doble

Interfaz:

```
void  WriteDouble(Double value);
```

Convertir un número de coma flotante de precisión doble a un entero largo y escribir el entero largo en la corriente de datos de mensaje como 8 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El número de coma flotante de precisión doble que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteFloat - Escribir número de coma flotante

Interfaz:

```
void  WriteFloat(Single value);
```

Convertir una número de coma flotante a un entero y escribir el entero en la corriente de datos de mensaje como 4 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El número de coma flotante que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteInt - Escribir entero

Interfaz:

```
void  WriteInt(Int32 value);
```

Escribir un entero en la corriente de datos de mensaje como 4 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteLong - Escribir entero largo

Interfaz:

```
void  WriteLong(Int64 value);
```

Escribir un entero largo en la corriente de datos de mensaje como 8 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero largo que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteObject - Escribir objeto

Interfaz:

```
void  WriteObject(Object value);
```

Escribir un valor, con un tipo de datos especificado, en la corriente de datos de mensaje.

Parámetros:

objectType (entrada)
El valor, que debe adoptar uno de los tipos de objeto siguientes:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

value (entrada)
Una matriz de bytes que contiene el valor que se va a escribir.

length (entrada)
El número de bytes de la matriz.

Devuelve:
Void

Excepciones:

- XMSException

WriteShort - Escribir entero corto

Interfaz:

```
void  WriteShort(Int16 value);
```

Escribir un entero corto en la corriente de datos de mensaje como 2 bytes, primero el byte de orden superior.

Parámetros:

value (entrada)
El entero corto que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

WriteString - Escribir serie

Interfaz:

```
void  WriteString(String value);
```

Escribir serie en la corriente de datos de mensaje.

Parámetros:

value (entrada)
Un objeto String que encapsula la serie que se va a escribir.

Devuelve:
Void

Excepciones:

- XMSException
- MessageNotWritableException

Propiedades y métodos heredados

Las propiedades siguientes se heredan de la interfaz IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Los métodos siguientes se heredan de la interfaz IMessage:

clearBody, clearProperties, PropertyExists

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

ITextMessage

Un mensaje de texto es un mensaje cuyo cuerpo está formado por una serie.

Jerarquía de herencia:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.ITextMessage
```

Referencia relacionada

Mensajes de texto
El cuerpo de un mensaje de texto contiene una serie.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
Text	Obtener y establecer la serie que forma el cuerpo del mensaje de texto.

Text - Obtener y establecer texto

Interfaz:

```
String Text
{
  get;
  set;
}
```

Obtener y establecer la serie que forma el cuerpo del mensaje de texto.

Si es necesario, XMS convierte los caracteres de la serie en la página de códigos local.

Excepciones:

- XMSException
- MessageNotReadableException
- MessageNotWritableException
- MessageEOFException

Propiedades y métodos heredados

Las propiedades siguientes se heredan de la interfaz IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Los métodos siguientes se heredan de la interfaz IMessage:

clearBody, clearProperties, PropertyExists

Los métodos siguientes se heredan de la interfaz IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty,

SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

TransactionInProgressException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.TransactionInProgressException
```

XMS lanza esta excepción si una aplicación solicita una operación que no es válida porque una transacción está en curso.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

TransactionRolledBackException

Jerarquía de herencia:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.TransactionRolledBackException
```

XMS lanza esta excepción si una aplicación llama a Session.commit() para confirmar la transacción actual, pero la transacción se ha retrotraído.

Propiedades y métodos heredados

Los métodos siguientes se heredan de la interfaz XMSException:

GetErrorCode, GetLinkedException

XMSException

Si XMS detecta un error al procesar una llamada a un método .NET, XMS lanza una excepción. Una excepción es un objeto que encapsula información sobre el error.

Jerarquía de herencia:

```
System.Exception
   |
   +----IBM.XMS.XMSException
```

Existen distintos tipos de excepción XMS, y un objeto XMSException es simplemente un tipo de excepción. Sin embargo, la clase XMSException es una superclase de las otras clases de excepción XMS. XMS lanza un objeto XMSException en situaciones donde ninguno de los otros tipos de excepción es apropiado.

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
ErrorCode	Obtener el código de error.
LinkedException	Obtener la siguiente excepción de la cadena de excepciones.

ErrorCode - Obtener código de error

Interfaz:

```
public String ErrorCode
{
  get {return errorCode_;}
}
```

Obtener el código de error.

Excepciones:

- XMSException

LinkedException - Obtener excepción enlazada

Interfaz:

```
public Exception LinkedException
{
  get { return linkedException_;}
  set { linkedException_ = value;}
}
```

Obtener la siguiente excepción de la cadena de excepciones.

El método devuelve un nulo, si no hay más excepciones en la cadena.

Excepciones:

- XMSException

XMSFactoryFactory

Si una aplicación no está utilizando objetos administrados, utilice esta clase para crear fábricas de conexiones, colas y temas.

Jerarquía de herencia:
Ninguna

Propiedades de .NET

Resumen de propiedades de .NET:

Propiedad .NET	Descripción
MetaData	Obtener los metadatos que son apropiados para el tipo de conexión del objeto XMSFactoryFactory.

Metadata - Recuperar metadatos

Interfaz:

```
IConnectionMetaData MetaData
```

Obtener los metadatos que son apropiados para el tipo de conexión del objeto XMSFactoryFactory.

Excepciones:
Ninguna

Métodos

Resumen de métodos:

Método	Descripción
CreateConnectionFactory	Crear un objeto ConnectionFactory del tipo declarado.
CreateQueue	Crear un objeto Destination para representar una cola en el servidor de mensajería.
CreateTopic	Crear un objeto Destination para representar un tema.
GetInstance	Cree una instancia de XMSFactoryFactory. Una aplicación XMS utiliza un objeto XMSFactoryFactory para obtener una referencia a un objeto ConnectionFactory que es apropiado para el tipo de protocolo necesario. Este objeto ConnectionFactory puede generar conexiones solo para ese tipo de protocolo.

CreateConnectionFactory - Crear fábrica de conexiones

Interfaz:

```
IConnectionFactory CreateConnectionFactory();
```

Crear un objeto ConnectionFactory del tipo declarado.

Parámetros:
Ninguna

Devuelve:
El objeto ConnectionFactory.

Excepciones:

- XMSException

CreateQueue - Crear cola

Interfaz:

```
IDestination CreateQueue(String name);
```

Crear un objeto Destination para representar una cola en el servidor de mensajería.

Este método no crea la cola en el servidor de mensajería. Debe crear la cola antes de que una aplicación pueda llamar a este método.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre de la cola, o encapsula un identificador uniforme de recursos (URI) que identifica la cola.

Devuelve:
El objeto Destination que representa la cola.

Excepciones:

- XMSException

CreateTopic - Crear tema

Interfaz:

```
IDestination CreateTopic(String name);
```

Crear un objeto Destination para representar un tema.

Parámetros:

name (entrada)
Un objeto String que encapsula el nombre del tema, o encapsula un identificador uniforme de recursos (URI) que identifica el tema.

Devuelve:
El objeto Destination que representa el tema.

Excepciones:

- XMSException

GetInstance - Obtener una instancia de XMSFactoryFactory

Interfaz:

```
static XMSFactoryFactory GetInstance(int connectionType);
```

Cree una instancia de XMSFactoryFactory. Una aplicación XMS utiliza un objeto XMSFactoryFactory para obtener una referencia a un objeto ConnectionFactory que es apropiado para el tipo de protocolo necesario. Este objeto ConnectionFactory puede generar conexiones solo para ese tipo de protocolo.

Parámetros:

connectionType (entrada)
El tipo de conexión para el cual el objeto ConnectionFactory genera conexiones.

- XMSC.CT_WPM
- XMSC.CT_RTT
- XMSC.CT_WMQ

Devuelve:
El objeto XMSFactoryFactory dedicado al tipo de conexión declarado.

Excepciones:

- NotSupportedException

Propiedades de objetos XMS

Este secciónCapítulo documenta las propiedades de objeto definidas por XMS.

El secciónCapítulo contiene los temassecciones siguientes:

- "Propiedades de conexión" en la página 184
- "Propiedades de ConnectionFactory" en la página 185
- "Propiedades de ConnectionMetaData" en la página 191
- "Propiedades de Destination" en la página 191
- "Propiedades de InitialContext" en la página 194
- "Propiedades de Message" en la página 195
- "Propiedades de MessageConsumer" en la página 200
- "Propiedades de MessageProducer" en la página 200
- "Propiedades de sesión" en la página 200

Cada temasección lista las propiedades de un objeto del tipo especificado y proporciona una breve descripción de cada propiedad.

Este secciónCapítulo también contiene el "Definiciones de propiedad" en la página 201 temasección, que proporciona una definición de cada propiedad.

Si una aplicación define sus propias propiedades de los objetos descritos en este secciónCapítulo, no provoca un error, pero podría generar resultados imprevisibles.

Nota: Los nombres y valores de propiedad de esta sección se muestran en el formato XMSC.*NAME*, que es el formato utilizado para C y C++. Sin embargo, en .NET, el formato del nombre de propiedad puede ser XMSC.*NAME* o XMSC_*NAME*, en función de cómo lo esté utilizando:

- Si está especificando una propiedad, el nombre de propiedad debe tener el formato XMSC.*NAME* tal como se muestra en el ejemplo siguiente:

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- Si está especificando una serie, el nombre de propiedad debe tener el formato XMSC_*NAME* tal como se muestra en el ejemplo siguiente:

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

  En .NET, los nombres y valores de propiedad se proporcionan como constantes en la clase XMSC. Estas constantes identifican series y las utilizaría cualquier aplicación de XMS .NET . Si está utilizando estas constantes predefinidas. los nombres y valores de propiedad tienen el formato XMSC.*NAME*, de modo que, por ejemplo, utilizaría XMSC.USERID, en lugar de XMSC_USERID.

Los tipos de datos también están en el formato utilizado para C/C++. Puede encontrar los valores correspondientes para .NET en"Tipos de datos para .NET" en la página 47.

Conceptos relacionados

Creación de sus propias aplicaciones
Cree sus propias aplicaciones como crea las aplicaciones de ejemplo.

Referencia relacionada

Interfaces .NET
En este temasección se documentan las interfaces de clase .NET y sus propiedades y métodos.

Propiedades de conexión

Una descripción general de las propiedades del objeto Connection, con enlaces a información de referencia más detallada.

Tabla 28. Propiedades de conexión

Nombre de la propiedad	Descripción
"XMSC_WMQ_RESOLVED_QUEUE_MANAGER" en la página 237	Esta propiedad se utiliza para obtener el nombre del gestor de colas que está conectado.
"XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID" en la página 237	Esta propiedad se llena con el ID del gestor de colas después de la conexión.
XMSC_WPM_CONNECTION_PROTOCOL	El protocolo de comunicaciones utilizado para la conexión al motor de mensajería. Esta propiedad es de solo lectura.
XMSC_WPM_HOST_NAME	El nombre de host o la dirección IP del sistema que contiene el motor de mensajería al que se conecta la aplicación. Esta propiedad es de solo lectura.
XMSC_WPM_ME_NAME	El nombre del motor de mensajería al que se conecta la aplicación. Esta propiedad es de solo lectura.
XMSC_WPM_PORT	El número del puerto en el que estaba a la escucha el motor de mensajería al que está conectada la aplicación. Esta propiedad es de solo lectura.

Un objeto Connection también tiene prioridad de solo lectura que se han derivado de las propiedades de la fábrica de conexiones que se utilizó para crear la conexión. Estas propiedades no solo se han derivado de las propiedades de fábrica de conexiones que se establecieron en el momento cuando se creó la conexión, sino que también de los valores predeterminados de las propiedades que no se establecieron. Las propiedades solo incluyen las que son relevantes para el tipo de servidor de mensajería al que

está conectada la aplicación. Los nombres de las propiedades son las mismos que los nombres de las propiedades de fábrica de conexiones.

Propiedades de ConnectionFactory

Una descripción general de las propiedades del objeto ConnectionFactory, con enlaces a información de referencia más detallada.

Tabla 29. Propiedades de ConnectionFactory

Nombre de la propiedad	Descripción
"XMSC_ASYNC_EXCEPTIONS" en la página 213	Esta propiedad determina si XMS informa de un ExceptionListener solo cuando se interrumpe una conexión, o cuando se produce cualquier excepción de forma asíncrona en una llamada de la API XMS. Esta propiedad se aplica a todas las conexiones creadas a partir de esta ConnectionFactory que tienen un ExceptionListener registrado.
XMSC_CLIENT_ID	El identificador de cliente para una conexión.
XMSC_CONNECTION_TYPE	El tipo del servidor de mensajería al que se conecta una aplicación.
XMSC_PASSWORD	Una contraseña que se puede utilizar para autenticar la aplicación cuando intenta conectarse a un servidor de mensajería.
"XMSC_RTT_BROKER_PING_INTERVAL" en la página 218	El intervalo de tiempo, en milisegundos, transcurrido el cual XMS .NET comprueba la conexión con un servidor de mensajería en tiempo real para detectar cualquier actividad.
XMSC_RTT_CONNECTION_PROTOCOL	El protocolo de comunicación utilizado para una conexión en tiempo real con un intermediario.
XMSC_RTT_HOST_NAME	El nombre de host o la dirección IP del sistema en el que se ejecuta un intermediario.
XMSC_RTT_LOCAL_ADDRESS	El nombre de host o la dirección IP de la interfaz de red local que se va a utilizar para una conexión en tiempo real con un intermediario.
XMSC_RTT_MULTICAST	El valor de multidifusión para una fábrica de conexiones o destino.
XMSC_RTT_PORT	El número del puerto en el cual un intermediario escucha solicitudes entrantes.
XMSC_USERID	Un identificador de usuario que se puede utilizar para autenticar la aplicación cuando intenta conectarse a un servidor de mensajería.

Tabla 29. Propiedades de ConnectionFactory (continuación)

Nombre de la propiedad	Descripción
XMSC_WMQ_BROKER_CONTROLQ	El nombre del gestor de colas utilizado por un intermediario. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_BROKER_PUBQ	El nombre de la cola supervisada por un intermediario donde las aplicaciones envían mensajes que publican. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_BROKER_QMGR	El nombre del gestor de colas al que está conectado un intermediario. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_BROKER_SUBQ	El nombre de la cola de suscriptor para un consumidor de mensajes no duraderos. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_BROKER_VERSION	El tipo de intermediario utilizado por la aplicación para una conexión o para el destino. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.

Tabla 29. Propiedades de ConnectionFactory (continuación)

Nombre de la propiedad	Descripción
"XMSC_WMQ_CCDTURL" en la página 223	Un localizador uniforme de recursos (URL) que identifica el nombre y la ubicación del archivo que contiene la tabla de la definición de canal de cliente y, también, especifica cómo se puede acceder al archivo.
XMSC_WMQ_CHANNEL	El nombre del canal que se va a utilizar para una conexión.
"XMSC_WMQ_CLIENT_RECONNECT_OPTIONS" en la página 224	Esta propiedad especifica las opciones de reconexión de cliente para nuevas conexiones creadas por esta fábrica
"XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT" en la página 224	Esta propiedad especifica la duración de tiempo, en segundos, que una conexión de cliente intenta reconectarse.
XMSC_WMQ_CONNECTION_MODE	La modalidad a través de la cual una aplicación se conecta a un gestor de colas.
"XMSC_WMQ_CONNECTION_NAME_LIST" en la página 225	Esta propiedad especifica los hosts a los que el cliente intenta reconectarse después de que se hayan interrumpido sus conexiones.
XMSC_WMQ_FAIL_IF_QUIESCE	Indica si las llamadas a determinados métodos fallan si el gestor de colas al que está conectada la aplicación está en un estado de inmovilización.
XMSC_WMQ_HOST_NAME	El nombre de host o la dirección IP del sistema en el cual se ejecuta un gestor de colas.
XMSC_WMQ_LOCAL_ADDRESS	Para una conexión a un gestor de colas, esta propiedad especifica la interfaz de red local que se va a utilizar, o el puerto local o el rango de puertos locales que se van a utilizar, o ambos.
XMSC_WMQ_MESSAGE_SELECTION	Determina si la selección de mensajes ha sido realizada por el cliente de XMS o el intermediario. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_MSG_BATCH_SIZE	El número máximo de mensajes que se va a recuperar de una cola en un lote cuando se utiliza la entrega de mensajes asíncrona. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.

Tabla 29. Propiedades de ConnectionFactory (continuación)

Nombre de la propiedad	Descripción
XMSC_WMQ_POLLING_INTERVAL	Si cada escucha de mensajes dentro de una sesión no tiene ningún mensaje adecuado en su cola, este valor es el intervalo máximo, en milisegundos, que transcurre antes de que cada escucha de mensajes intente de nuevo obtener un mensaje de su cola. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
"XMSC_WMQ_PROVIDER_VERSION" en la página 234	La versión, release, nivel de modificación y fixpack del gestor de colas al que tiene intención conectarse la aplicación.
XMSC_WMQ_PORT	El número del puerto en el cual un gestor de colas escucha solicitudes entrantes.
XMSC_WMQ_PUB_ACK_INTERVAL	Número de mensajes publicados por una aplicación de publicación antes de que el cliente XMS solicite un acuse de recibo al intermediario. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
"XMSC_WMQ_PUT_ASYNC_ALLOWED" en la página 229	Esta propiedad determina si los productores de mensajes están permitidos para utilizar operaciones de transferencia asíncrona para enviar mensajes a este destino.
XMSC_WMQ_QMGR_CCSID	El identificador (CCSID) del juego de caracteres codificado, o página de códigos, en el que se intercambian los campos de datos de caracteres definidos en la interfaz de cola de mensajes (MQI) entre el cliente de XMS y el cliente de IBM WebSphere MQ .
XMSC_WMQ_QUEUE_MANAGER	El nombre del gestor de colas al que conectarse.
XMSC_WMQ_RECEIVE_EXIT	Identifica una salida de recepción de canal que se va a ejecutar.
XMSC_WMQ_RECEIVE_EXIT_INIT	Los datos de usuario que se pasan a una salida de recepción de canal cuando se llama.
XMSC_WMQ_SECURITY_EXIT	Identifica una salida de seguridad de canal.
XMSC_WMQ_SECURITY_EXIT_INIT	Los datos de usuario que se pasan a una salida de seguridad de canal cuando se llama.
"XMSC_WMQ_SEND_CHECK_COUNT" en la página 238	El número de llamadas de envío que se van a permitir entre las comprobaciones de errores de colocación asíncrona dentro de una sesión XMS única sin transacción.

Tabla 29. Propiedades de ConnectionFactory (continuación)	
Nombre de la propiedad	**Descripción**
XMSC_WMQ_SEND_EXIT	Identifica una salida de emisión de canal.
XMSC_WMQ_SEND_EXIT_INIT	Los datos de usuario que se pasan a las salidas de emisión de canal cuando se invocan.
"XMSC_WMQ_SHARE_CONV_ALLOWED" en la página 239	Indica si una conexión de cliente puede compartir su socket con otras conexiones XMS de nivel superior del mismo proceso con el mismo gestor de colas, si las definiciones de canal coinciden. Esta propiedad se proporciona para permitir un aislamiento completo de conexiones en sockets separados si es necesario para el desarrollo de aplicaciones, el mantenimiento o por motivos operativos.
XMSC_WMQ_SSL_CERT_STORES	Las ubicaciones de los servidores que contienen las listas de revocaciones de certificados (CRL) que se van a utilizar en una conexión SSL a un gestor de colas.
XMSC_WMQ_SSL_CIPHER_SPEC	El nombre de la CipherSpec que se va a utilizar en una conexión segura a un gestor de colas.
XMSC_WMQ_SSL_CIPHER_SUITE	El nombre de la CipherSuite que se va a utilizar en una conexión SSL o TLS con un gestor de colas. El protocolo utilizado en la negociación de la conexión segura depende de la CipherSuite especificada.
XMSC_WMQ_SSL_CRYPTO_HW	Detalles de configuración para el hardware de cifrado conectado al sistema cliente.
XMSC_WMQ_SSL_FIPS_REQUIRED	El valor de esta propiedad determina si una aplicación puede o no puede utilizar suites de cifrado no compatibles con FIPS. Si esta propiedad se establece en true (verdadero), solo se utilizan algoritmos FIPS para la conexión cliente/servidor.
XMSC_WMQ_SSL_KEY_REPOSITORY	La ubicación de un archivo de base de datos de claves en el cual se almacenan claves y certificados.
XMSC_WMQ_SSL_KEY_RESETCOUNT	El KeyResetCount representa el número total de bytes sin cifrado enviados y recibidos en una conversación SSL antes de renegociar la clave secreta.
XMSC_WMQ_SSL_PEER_NAME	El nombre de igual que se va a utilizar en una conexión SSL a un gestor de colas.
XMSC_WMQ_SYNCPOINT_ALL_GETS	Indica si se deben recuperar todos los mensajes de colas dentro del control de punto de sincronización.
"XMSC_WMQ_TARGET_CLIENT" en la página 246	
XMSC_WMQ_TEMP_Q_PREFIX	El prefijo utilizado para formar el nombre de la cola dinámica IBM WebSphere MQ que se crea cuando la aplicación crea la cola temporal Un XMS .

Tabla 29. Propiedades de ConnectionFactory (continuación)	
Nombre de la propiedad	**Descripción**
XMSC_WMQ_TEMP_TOPIC_PREFIX	Al crear temas temporales, XMS genera una serie de tema con el formato "TEMP/TEMPTOPICPREFIX/unique_id", o si esta propiedad se deja con el valor predeterminado, sólo "TEMP/unique_id". Si se especifica un valor no vacío se permite que se definan colas de modelo específicas para crear las colas gestionadas para suscriptores de temas temporales creados en esta conexión.
XMSC_WMQ_TEMPORARY_MODEL	Nombre de la cola modelo de IBM WebSphere MQ a partir de la cual se crea una cola dinámica cuando la aplicación crea una cola temporal de Un XMS .
XMSC_WPM_BUS_NAME	Para una fábrica de conexiones, el nombre del bus de integración de servicios al que se conecta la aplicación o, para un destino, el nombre del bus de integración de servicios en el cual existe el destino.
XMSC_WPM_CONNECTION_PROXIMITY	El valor de proximidad de conexión para la conexión.
XMSC_WPM_DUR_SUB_HOME	El nombre del motor de mensajería donde se gestionan todas las suscripciones duraderas para una conexión o destino.
XMSC_WPM_LOCAL_ADDRESS	Para una conexión a un bus de integración de servicios, esta propiedad especifica la interfaz de red local que se va a utilizar, o el puerto local o el rango de puertos locales que se va a utilizar, o ambos.
XMSC_WPM_NON_PERSISTENT_MAP	El nivel de fiabilidad de los mensajes no persistente que se envían utilizando la conexión.
XMSC_WPM_PERSISTENT_MAP	El nivel de fiabilidad de los mensajes persistentes que se envían mediante la conexión.
XMSC_WPM_PROVIDER_ENDPOINTS	Una secuencia de una o más direcciones de punto final de servidores de programa de arranque.
XMSC_WPM_TARGET_GROUP	El nombre de un grupo de destino de motores de mensajería.
XMSC_WPM_TARGET_SIGNIFICANCE	La importancia del grupo de destinos de motores de mensajería.
XMSC_WPM_TARGET_TRANSPORT_CHAIN	El nombre de la cadena de transporte de entrada que debe utilizar la aplicación para conectarse a un motor de mensajería.
XMSC_WPM_TARGET_TYPE	El tipo del grupo de destinos de motores de mensajería.
XMSC_WPM_TEMP_Q_PREFIX	El prefijo utilizado para formar el nombre de la cola temporal que se crea en el bus de integración de servicios cuando la aplicación crea la cola temporal de Un XMS.
XMSC_WPM_TEMP_TOPIC_PREFIX	El prefijo utilizado para formar el nombre de un tema temporal que crea la aplicación.

Conceptos relacionados

Objetos ConnectionFactories y Connection

Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Conexiones seguras a un IBM WebSphere MQ gestor de colas
Para habilitar una aplicación XMS .NET para realizar conexiones seguras con un IBM WebSphere MQ gestor de colas, las propiedades relevantes deben estar definidas en el objeto ConnectionFactory .

Conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus
Para permitir que una aplicación XMS de .NET establezca conexiones seguras con un motor de mensajería de WebSphere Servicio Integration Bus, se deben definir las propiedades pertinentes en el objeto ConnectionFactory.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

Propiedades necesarias para objetos ConnectionFactory administrados
Cuando una aplicación crea una fábrica de conexiones, se debe definir un número de propiedades para crear una conexión a un servidor de mensajería.

Propiedades de ConnectionMetaData

Una descripción general de las propiedades del objeto ConnectionMetaData, con enlaces a información de referencia más detallada.

Tabla 30. Propiedades de ConnectionMetaData

Nombre de la propiedad	Descripción
XMSC_JMS_MAJOR_VERSION	El número de versión principal de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.
XMSC_JMS_MINOR_VERSION	El número de versión menor de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.
XMSC_JMS_VERSION	El identificador de versión de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.
XMSC_MAJOR_VERSION	El número de versión del cliente XMS. Esta propiedad es de solo lectura.
XMSC_MINOR_VERSION	El número de release del cliente XMS. Esta propiedad es de solo lectura.
XMSC_PROVIDER_NAME	El proveedor del cliente XMS. Esta propiedad es de solo lectura.
XMSC_VERSION	El identificador de versión del cliente XMS. Esta propiedad es de solo lectura.

Propiedades de Destination

Una descripción general de las propiedades del objeto Destination, con enlaces a información de referencia a más detallada.

Tabla 31. Propiedades de Destination	
Nombre de la propiedad	**Descripción**
XMSC_DELIVERY_MODE	La modalidad de entrega de mensajes enviados al destino.
XMSC_PRIORITY	La prioridad de los mensajes enviados al destino.
XMSC_RTT_MULTICAST	El valor de multidifusión para una fábrica de conexiones o destino.
XMSC_TIME_TO_LIVE	El tiempo de vida para los mensajes enviados al destino.
XMSC_WMQ_BROKER_VERSION	El tipo de intermediario utilizado por la aplicación para una conexión o para el destino.
XMSC_WMQ_CCSID	El identificador (CCSID) del juego de caracteres codificado, o página de códigos, en el que están escritas las series de datos de carácter en el cuerpo de un mensaje cuando el cliente de XMS envía el mensaje al destino.
XMSC_WMQ_DUR_SUBQ	El nombre de la cola de suscriptor para un suscriptor duradero que está recibiendo mensajes del destino. **Nota:** Esta propiedad se puede utilizar con la versión 2.0 de IBM Message Service Client for .NET pero no tiene ningún efecto para una aplicación conectada a un gestor de colas IBM WebSphere MQ Versión 7.0, a menos que la propiedad XMSC_WMQ_PROVIDER_VERSION de la fábrica de conexiones esté establecida en un número de versión inferior a 7.
XMSC_WMQ_ENCODING	Cómo se representan datos numéricos en el cuerpo de un mensaje cuando el cliente XMS envía el mensaje al destino.
XMSC_WMQ_FAIL_IF_QUIESCE	Indica si las llamadas a determinados métodos fallan si el gestor de colas al que está conectada la aplicación está en un estado de inmovilización.
“XMSC_WMQ_MESSAGE_BODY” en la página 227	Esta propiedad determina si una aplicación XMS procesa la MQRFH2 de un mensaje IBM WebSphere MQ como parte de la carga útil del mensaje (es decir, como parte del cuerpo del mensaje).
“XMSC_WMQ_MQMD_MESSAGE_CONTEXT” en la página 228	Determina qué nivel de contexto de mensaje debe establecer la aplicación XMS . La aplicación debe estar en ejecución con la autoridad de contexto adecuada para que esta propiedad tenga efecto.
“XMSC_WMQ_MQMD_READ_ENABLED” en la página 229	Esta propiedad determina si una aplicación XMS puede extraer los valores de los campos MQMD o no.
“XMSC_WMQ_MQMD_WRITE_ENABLED” en la página 229	Esta propiedad determina si una aplicación XMS puede indicar los valores de los campos MQMD o no.
“XMSC_WMQ_READ_AHEAD_ALLOWED” en la página 230	Esta propiedad determina si los consumidores de mensajes y los navegadores de colas están autorizados para utilizar la lectura anticipada para obtener mensajes no persistentes y no transaccionales de este destino en un almacenamiento intermedio interno antes de recibirlos.

Tabla 31. Propiedades de Destination (continuación)	
Nombre de la propiedad	**Descripción**
"XMSC_WMQ_READ_AHEAD_CLOSE_POLICY" en la página 230	Esta propiedad determina, para los mensajes que se están entregando a un escucha de mensajes asíncronos, qué sucede con los mensajes en el almacenamiento de lectura anticipada interno, cuando se cierra el consumidor de mensajes.
"XMSC_WMQ_RECEIVE_CCSID" en la página 236	La propiedad de destino que define el CCSID de destino para la conversión de mensajes del gestor de colas. El valor se ignora, a menos que XMSC_WMQ_RECEIVE_CONVERSION se establezca en WMQ_RECEIVE_CONVERSION_QMGR.
"XMSC_WMQ_RECEIVE_CONVERSION" en la página 236	La propiedad de destino que determina si el gestor de colas va a realizar la conversión de datos.
XMSC_WMQ_TARGET_CLIENT	Indica si los mensajes enviados al destino contienen una cabecera MQRFH2.
XMSC_WMQ_TEMP_TOPIC_PREFIX	Al crear temas temporales, XMS genera una serie de tema con el formato "`TEMP/TEMPTOPICPREFIX/unique_id`", o si esta propiedad se deja con el valor predeterminado, sólo "`TEMP/unique_id`". Si se especifica un valor no vacío se permite que se definan colas de modelo específicas para crear las colas gestionadas para suscriptores de temas temporales creados en esta conexión.
XMSC_WPM_BUS_NAME	Para una fábrica de conexiones, el nombre del bus de integración de servicios al que se conecta la aplicación o, para un destino, el nombre del bus de integración de servicios en el cual existe el destino.
XMSC_WPM_TOPIC_SPACE	El nombre del espacio de tema que contiene el tema.

Conceptos relacionados

Objetos ConnectionFactories y Connection
Un objeto ConnectionFactory proporciona una plantilla que utiliza una aplicación para crear un objeto Connection. La aplicación utiliza el objeto Connection para crear un objeto Session.

Conexión con un bus de integración de servicios WebSphere
La aplicación Un XMS puede conectar con un WebSphere Servicio Integration Bus, ya sea utilizando una conexión TCP/IP directa o utilizando HTTP sobre TCP/IP.

Destinos
Una aplicación XMS utiliza un objeto Destination para especificar el destino de mensajes que se están enviando y el origen de mensajes que se están recibiendo.

Comodines de destino
XMS proporciona soporte para comodines de destino, garantizando que los comodines se pueden pasar hasta el lugar donde son necesarios para la coincidencia. Hay un esquema de comodín diferente para cada tipo de servidor con el que puede trabajar XMS.

Identificadores uniformes de recursos de tema
El identificador uniforme de recursos (URI) de tema especifica el nombre del tema; también puede especificar una o más propiedades para el mismo.

Identificadores uniformes de recursos de cola
El URI de una cola especifica el nombre de la cola; también puede especificar una o más propiedades de la cola.

Destinos temporales

Las aplicaciones XMS pueden crear y utilizar destinos temporales.

Correlación de propiedades para objetos administrados
Para permitir que las aplicaciones utilicen definiciones de objeto de destino y de fábrica de conexiones de JMS IBM WebSphere MQ y de WebSphere Application Server, las propiedades recuperadas a partir de estas definiciones se deben correlacionar con las propiedades XMS correspondientes que se pueden establecer en los destinos y las fábricas de conexiones de XMS.

Tareas relacionadas

Creación de objetos administrados
Las definiciones de los objetos ConnectionFactory y Destination que necesitan las aplicaciones XMS para establecer una conexión a un servidor de mensajería se deben crear utilizando las herramientas administrativas apropiadas.

Referencia relacionada

Propiedades necesarias para objetos Destination administrados
Una aplicación que está creando un destino debe establecer varias propiedades para la aplicación en un objeto Destination administrado.

Propiedades de InitialContext

Una descripción general de las propiedades del objeto InitialContext, con enlaces a información de referencia más detallada.

Tabla 32. Propiedades de InitialContext

Nombre de la propiedad	Descripción
XMSC_IC_URL	Para contextos de LDAP y FileSystem, la dirección del repositorio que contiene objetos administrados.

Tabla 33. Propiedades de InitialContext

Nombre de la propiedad	Descripción
XMSC_IC_PROVIDER_URL	Se utiliza para localizar el directorio de denominación JNDI de forma que no es necesario que el servicio de denominación COS esté en el mismo servidor que el servicio web.
XMSC_IC_SECURITY_AUTHENTICATION	Basado en la interfaz de contexto Java SECURITY_AUTHENTICATION. Esta propiedad solo es aplicable al contexto de denominación COS.
XMSC_IC_SECURITY_CREDENTIALS	Basado en la interfaz de contexto Java SECURITY_CREDENTIALS. Esta propiedad solo es aplicable al contexto de denominación COS.
XMSC_IC_SECURITY_PRINCIPAL	Basado en la interfaz de contexto Java SECURITY_PRINCIPAL. Esta propiedad solo es aplicable al contexto de denominación COS.
XMSC_IC_SECURITY_PROTOCOL	Basado en la interfaz de contexto Java SECURITY_PROTOCOL Esta propiedad solo es aplicable al contexto de denominación COS.
XMSC_IC_URL	Para contextos de LDAP y FileSystem, la dirección del repositorio que contiene objetos administrados. Para contextos de denominación COS, la dirección del servicio web que busca los objetos en el directorio.

Conceptos relacionados

Propiedades InitialContext
Los parámetros del constructor InitialContext incluyen la ubicación del repositorio de objetos administrados, dada como un identificador uniforme de recursos (URI). Para que una aplicación pueda establecer una conexión con el repositorio, puede ser necesario proporcionar más información que la información incluida en el URI.

Formato URI para contextos iniciales de XMS
La ubicación del repositorio de objetos administrados se proporciona como un identificador uniforme de recursos (URI). El formato del URI depende del tipo de contexto.

Recuperación de objetos administrados
XMS recupera un objeto administrado del repositorio utilizando la dirección proporcionada cuando se crea el objeto InitialContext, o en las propiedades InitialContext.

Tareas relacionadas

Objetos InitialContext
Una aplicación debe crear un contexto inicial que se va a utilizar para realizar una conexión con el repositorio de objetos administrados para recuperar los objetos administrados necesarios.

Propiedades de Message

Una descripción general de las propiedades del objeto Message, con enlaces a información de referencia más detallada.

Tabla 34. Propiedades de Message

Nombre de la propiedad	Descripción
JMS_IBM_CHARACTER_SET	El identificador (CCSID) del juego de caracteres codificado, la página de códigos, en la que se escriben las series de datos de carácter en el cuerpo del mensaje cuando el cliente XMS envía el mensaje a su destino previsto. En XMS, esta propiedad tiene un valor numérico y se correlaciona con CCSID. Si embargo, esta propiedad se basa en una propiedad JMS de modo que tiene un valor de tipo serie y se correlaciona con el juego de caracteres Java que representa este CCSID numérico.
JMS_IBM_ENCODING	Cómo se representan los datos numéricos en el cuerpo del mensaje cuando el cliente XMS envía el mensaje a su destino previsto.
JMS_IBM_EXCEPTIONMESSAGE	Texto que describe la razón por la cual el mensaje se ha enviado al destino de excepciones. Esta propiedad es de solo lectura.
JMS_IBM_EXCEPTIONPROBLEMDESTINATION	El nombre del destino en el que estaba el mensaje antes de que se enviara al destino de excepciones.
JMS_IBM_EXCEPTIONREASON	Un código de razón que indica la razón por la cual el mensaje se envió al destino de excepciones.
JMS_IBM_EXCEPTIONTIMESTAMP	La hora cuando se envió el mensaje al destino de excepciones.
JMS_IBM_FEEDBACK	Un código que indica la naturaleza de un mensaje de informe.
JMS_IBM_FORMAT	La naturaleza de los datos de aplicación del mensaje.
JMS_IBM_LAST_MSG_IN_GROUP	Indica si el mensaje es el último mensaje de un grupo de mensajes.

Tabla 34. Propiedades de Message (continuación)	
Nombre de la propiedad	**Descripción**
JMS_IBM_MSGTYPE	El tipo del mensaje.
JMS_IBM_PUTAPPLTYPE	El tipo de la aplicación que envió el mensaje.
JMS_IBM_PUTDATE	La fecha cuando se envió el mensaje.
JMS_IBM_PUTTIME	La hora cuando se envió el mensaje.
JMS_IBM_REPORT_COA	Solicitar los mensajes de informe 'confirmar en llegada', especificando cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.
JMS_IBM_REPORT_COD	Solicitar mensaje de informe 'confirmar en llegada', que especifica cuántos datos d e aplicación del mensaje original se deben incluir en un mensaje de informe.
JMS_IBM_REPORT_DISCARD_MSG	Solicitar que el mensaje se descarte si no se puede entregar a su destino previsto.
JMS_IBM_REPORT_EXCEPTION	Solicitar mensajes de informe de excepción, que especifican cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.
JMS_IBM_REPORT_EXPIRATION	Solicitar mensajes de informe de caducidad, que especifican cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.
JMS_IBM_REPORT_NAN	Solicitar mensajes de informe de notificación de acción negativa.
JMS_IBM_REPORT_PAN	Solicitar mensajes de informe de notificación de acción positiva.
JMS_IBM_REPORT_PASS_CORREL_ID	Solicite que el identificador de correlación de cualquier mensaje de informe o de respuesta sea el mismo que el identificador de correlación del mensaje original.
JMS_IBM_REPORT_PASS_MSG_ID	Solicite que el identificador de mensaje de cualquier mensaje de informe o respuesta sea el mismo que el identificador de mensaje del mensaje original.
JMS_IBM_RETAIN	Establecer esta propiedad indica al gestor de colas que trate un mensaje como una Publicación retenida.
JMS_IBM_SYSTEM_MESSAGEID	Un identificador que identifica el mensaje de forma exclusiva dentro del bus de integración de servicios. Esta propiedad es de solo lectura.
JMSX_APPID	El nombre de la aplicación que envió el mensaje.
JMSX_DELIVERY_COUNT	El número de intentos para entregar el mensaje.
JMSX_GROUPID	El identificador del grupo de mensajes al que pertenece el mensaje.
JMSX_GROUPSEQ	El número de secuencia del mensaje de un grupo de mensajes.
JMSX_USERID	El identificador de usuario asociado con la aplicación que envió el mensaje.

Propiedades JMS_IBM_MQMD*

IBM Message Service Client for .NET permite a las aplicaciones cliente leer/escribir campos MQMD utilizando distintas API. También permite el acceso a datos de mensaje MQ. De forma predeterminada, el acceso a MQMD está inhabilitado y la aplicación la debe habilitar de forma explícita utilizando las propiedades Destination XMSC_WMQ_MQMD_WRITE_ENABLED y XMSC_WMQ_MQMD_READ_ENABLED. Estas dos propiedades son independientes entre sí.

Todos los campos MQMD excepto StrucId y Version se exponen como propiedades de objeto Message adicionales y se añaden como prefijo a JMS_IBM_MQMD.

Las propiedades JMS_IBM_MQMD* tienen una mayor prioridad sobre otras propiedades como JMS_IBM* descritas en la tabla anterior.

Envío de mensajes

Están representados todos los campos MQMD excepto StrucId y Version. Estas propiedades solo hacen referencia a los campos MQMD; donde una propiedad se produce tanto en MQMD como en la cabeceraMQRFH2, la versión en MQRFH2 no se establece ni se extrae. Se puede establecer cualquiera de estas propiedades, excepto JMS_IBM_MQMD_BackoutCount. Se ignora cualquier valor establecido para JMS_IBM_MQMD_BackoutCount.

Si una propiedad tiene una longitud máxima y proporciona un valor que es demasiado largo, el valor se trunca.

Para determinadas propiedades, también debe establecer la propiedad XMSC_WMQ_MQMD_MESSAGE_CONTEXT en el objeto Destination. La aplicación debe estar en ejecución con la autoridad de contexto adecuada para que esta propiedad tenga efecto. Si no establece XMSC_WMQ_MQMD_MESSAGE_CONTEXT en un valor apropiado, se ignora el valor de propiedad. Si establece XMSC_WMQ_MQMD_MESSAGE_CONTEXT en un valor apropiado, pero no tiene suficiente autoridad de contexto para el gestor de colas, se emite una excepción. Las propiedades que requieren valores específicos de XMSC_WMQ_MQMD_MESSAGE_CONTEXT son las siguientes.

Las propiedades siguientes requieren que XMSC_WMQ_MQMD_MESSAGE_CONTEXT esté establecido en XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT o XMSC_WMQ_MDCTX_SET_ALL_CONTEXT:

- JMS_IBM_MQMD_UserIdentifier
- JMS_IBM_MQMD_AccountingToken
- JMS_IBM_MQMD_ApplIdentityData

Las propiedades siguientes requieren que XMSC_WMQ_MQMD_MESSAGE_CONTEXT esté establecido en XMSC_WMQ_MDCTX_SET_ALL_CONTEXT:

- JMS_IBM_MQMD_PutApplType
- JMS_IBM_MQMD_PutApplName
- JMS_IBM_MQMD_PutDate
- JMS_IBM_MQMD_PutTime
- JMS_IBM_MQMD_ApplOriginData

Recepción de mensajes

Todas estas propiedades están disponibles en un mensaje recibido si la propiedad XMSC_WMQ_MQMD_READ_ENABLED está establecida en true, independientemente de las propiedades reales que ha establecido la aplicación productora. Una aplicación no puede modificar las propiedades de un mensaje recibido, a menos que primero se borren todas las propiedades, de acuerdo con la especificación JMS. El mensaje recibido se puede enviar sin modificar las propiedades.

Nota: Si la aplicación recibe un mensaje de un destino con la propiedad XMSC_WMQ_MQMD_READ_ENABLED establecida en true, y lo envía a un destino con XMSC_WMQ_MQMD_WRITE_ENABLED establecido en true, esto genera que todos los valores de campo MQMD del mensaje recibido se copien en el mensaje enviado. Tabla de propiedades

Tabla 35. Propiedades del objeto Message que representa los campos MQMD		
Propiedad	**Descripción**	**Tipo**
JMS_IBM_MQMD_REPORT	Opciones para mensajes de informe	System.Int32
JMS_IBM_MQMD_MSGTYPE	Tipo de mensaje	System.Int32
JMS_IBM_MQMD_EXPIRY	Duración del mensaje	System.Int32
JMS_IBM_MQMD_FEEDBACK	Código de comentario o razón	System.Int32
JMS_IBM_MQMD_ENCODING	Codificación numérica de datos de mensaje	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	Identificador de juego de caracteres de datos de mensaje	System.Int32
JMS_IBM_MQMD_FORMAT	Nombre de formato de datos de mensaje	System.String
JMS_IBM_MQMD_PRIORITY **Nota:** Si asigna un valor a JMS_IBM_MQMD_PRIORITY que no está dentro del rango de 0 a 9, este valor infringe la especificación JMS.	Prioridad de mensaje	System.Int32
JMS_IBM_MQMD_PERSISTENCE	Persistencia de los mensajes	System.Int32
JMS_IBM_MQMD_MSGID **Nota:** La especificación JMS indica que el ID de mensaje debe ser establecido por el proveedor JMS y que se debe ser exclusivo o nulo. Si asigna un valor a JMS_IBM_MQMD_MSGID, este valor se copia en el JMSMessageID. De esta forma, el proveedor JMS no lo establece y podría no ser exclusivo: este valor infringe la especificación JMS.	Identificador del mensaje	Matriz de bytes **Nota:** El uso de las propiedades de matriz de bytes en un mensaje infringe la especificación JMS.
JMS_IBM_MQMD_CORRELID **Nota:** Si asigna un valor a JMS_IBM_MQMD_CORRELID que empieza con la serie 'ID:', este valor infringe la especificación JMS.	Identificador de correlación	Matriz de bytes **Nota:** El uso de las propiedades de matriz de bytes en un mensaje infringe la especificación JMS.
JMS_IBM_MQMD_BACKOUTCOUNT	Contador de restitución	System.Int32
JMS_IBM_MQMD_REPLYTOQ	Nombre de la cola de respuestas	System.String
JMS_IBM_MQMD_REPLYTOQMGR	Nombre del gestor de colas de respuestas	System.String
JMS_IBM_MQMD_USERIDENTIFIER	Identificador de usuario	System.String

Tabla 35. Propiedades del objeto Message que representa los campos MQMD (continuación)		
Propiedad	**Descripción**	**Tipo**
JMS_IBM_MQMD_ACCOUNTINGTOKEN	Señal de contabilidad	Matriz de bytes **Nota:** El uso de las propiedades de matriz de bytes en un mensaje infringe la especificación JMS.
JMS_IBM_MQMD_APPLIDENTITYDATA	Datos de aplicación relacionados con la identidad	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	Tipo de aplicación que coloca el mensaje	System.Int32
JMS_IBM_MQMD_PUTAPPLNAME	Nombre de la aplicación que coloca el mensaje	System.String
JMS_IBM_MQMD_PUTDATE	Fecha cuando se colocó el mensaje	System.String
JMS_IBM_MQMD_PUTTIME	Hora cuando se colocó el mensaje	System.String
JMS_IBM_MQMD_APPLORIGINDATA	Datos de aplicación relacionados con el origen	System.String
JMS_IBM_MQMD_GROUPID	Identificador de grupo	Matriz de bytes **Nota:** El uso de las propiedades de matriz de bytes en un mensaje infringe la especificación JMS.
JMS_IBM_MQMD_MSGSEQNUMBER	Número de secuencia del mensaje local dentro del grupo	System.Int32
JMS_IBM_MQMD_OFFSET	Desplazamiento de datos en el mensaje físico desde el inicio del mensaje lógico	System.Int32
JMS_IBM_MQMD_MSGFLAGS	Distintivos de mensajes	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	Longitud del mensaje original	System.Int32

Consulte MQMD si desea más detalles.

Ejemplos

Este ejemplo provoca que un mensaje se coloque en una cola o un tema con MQMD.UserIdentifier establecido en"JoeBloggs".

```
// Create a ConnectionFactory, connection, session, producer, message
// ...

// Create a destination
// ...

// Enable MQMD write
dest.setBooleanProperty(XMSC_WMQ_MQMD_WRITE_ENABLED,
      XMSC_WMQ_MQMD_WRITE_ENABLED_YES);

// Optionally, set a message context if applicable for this MD field
dest.setIntProperty(XMSC_WMQ_MQMD_MESSAGE_CONTEXT,
  XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT);
```

```
// On the message, set property to provide custom UserId
msg.setStringProperty(JMS_IBM_MQMD_USERIDENTIFIER, "JoeBloggs");

// Send the message
// ...
```

Es necesario establecer XMSC_WMQ_MQMD_MESSAGE_CONTEXT antes de establecer JMS_IBM_MQMD_USERIDENTIFIER. Si desea más información sobre el uso de XMSC_WMQ_MQMD_MESSAGE_CONTEXT, consulte las propiedades del objeto Message.

De forma similar, puede extraer el contenido de los campos MQMD estableciendo XMSC_WMQ_MQMD_READ_ENABLED en true antes de recibir un mensaje y, después, utilizar los métodos get del mensaje como, por ejemplo, getStringProperty. Las propiedades recibidas son de solo lectura.

Este ejemplo tiene como resultado que el campo de valor que contiene el valor del campo MQMD.ApplIdentityData de un mensaje se obtenga de una cola o de un tema.

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(XMSC_WMQ_MQMD_READ_ENABLED, XMSC_WMQ_MQMD_READ_ENABLED_YES);

// Receive a message
// ...

// Get desired MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

Propiedades de MessageConsumer

Una descripción general de las propiedades del objeto MessageConsumer, con enlaces a información de referencia más detallada.

Tabla 36. Propiedades de MessageConsumer

Nombre de la propiedad	Descripción
XMSC_IS_SUBSCRIPTION_MULTICAST	Indica si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport. Esta propiedad es de solo lectura.
XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST	Indica si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport con una calidad de servicio fiable. Esta propiedad es de solo lectura.

Consulte las Propiedades .NET de IMessageConsumer si desea más detalles.

Propiedades de MessageProducer

Una descripción general de las propiedades del objeto MessageProducer, con enlaces a información de referencia más detallada.

Consulte .Propiedades NET de IMessageProducer para obtener más detalles.

Propiedades de sesión

Una descripción general de las propiedades del objeto Session, con enlaces a información de referencia más detallada.

Consulte .Propiedades NET de ISession para obtener más detalles.

Definiciones de propiedad

Este temasección proporciona una definición de cada propiedad de objeto.

Cada definición de propiedad incluye la información siguiente:

- El tipo de datos de la propiedad
- Los tipos de objeto que tienen la propiedad
- Para una propiedad de Destination, el nombre que se puede utilizar en un identificador uniforme de recursos (URI)
- Una descripción más detallada de la propiedad
- Los valores válidos de la propiedad
- El valor predeterminado de la propiedad

Las propiedades cuyos nombres empiezan con uno de los prefijos siguientes solo son relevantes para el tipo de conexión especificado:

XMSC_RTT
Las propiedades solo son relevantes para una conexión en tiempo real con un intermediario. Los nombres de las propiedades están definidos como constantes con nombre en el archivo de cabecera `xmsc_rtt.h`.

XMSC_WMQ
Las propiedades solo son relevantes cuando una aplicación se conecta a un gestor de colas IBM WebSphere MQ. Los nombres de las propiedades están definidos como constantes con nombre en el archivo de cabecera `xmsc_wmq.h`.

XMSC_WPM
Las propiedades solo son relevantes cuando una aplicación se conecta a un bus de integración de servicios WebSphere. Los nombres de las propiedades están definidos como constantes con nombre en el archivo de cabecera `xmsc_wpm.h`.

A menos que se indique lo contrario en sus definiciones, las propiedades restantes son relevantes para todos los tipos de conexión. Los nombres de las propiedades están definidos como constantes con nombre en el archivo de cabecera `xmsc.h`. Las propiedades cuyos nombres empiezan con el prefijo JMSX son propiedades definidas de JMS de un mensaje, y las propiedades cuyos nombres empiezan con el prefijo JMS_IBM son las propiedades definidas de IBM de un mensaje. Si desea más información sobre las propiedades de mensajes, consulte “Propiedades de mensaje de Un XMS” en la página 74.

A menos que se indique de otra forma en su definición, cada propiedad es relevante en ambos dominios, punto a punto y Publicar/Suscribir.

Una aplicación puede obtener y establecer el valor de cualquier propiedad, a menos que la propiedad se haya designado como de solo lectura.

Están definidas las propiedades siguientes:

JMS_IBM_CHARACTER_SET

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

El identificador (CCSID) del juego de caracteres codificado, la página de códigos, en la que se escriben las series de datos de carácter en el cuerpo del mensaje cuando el cliente XMS envía el mensaje a su destino previsto. En XMS, esta propiedad tiene un valor numérico y se correlaciona con CCSID. Si embargo, esta propiedad se basa en una propiedad JMS de modo que tiene un valor de tipo serie y se correlaciona con el

juego de caracteres Java que representa este CCSID numérico. Esta propiedad sustituye cualquier CCSID especificada para el destino mediante la propiedad XMSC_WMQ_CCSID.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_ENCODING

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Cómo se representan los datos numéricos en el cuerpo del mensaje cuando el cliente XMS envía el mensaje a su destino previsto. Esta propiedad sustituye cualquier codificación especificada para el destino por la propiedad XMSC_WMQ_ENCODING. La propiedad especifica la representación de enteros binarios, enteros decimales empaquetados y números de coma flotante.

Los valores válidos de la propiedad son los mismos que los valores que se pueden especificar en el campo *Encoding* de un descriptor de mensaje. Para obtener más información sobre el campo *Encoding* , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

Una aplicación puede utilizar las constantes con nombre siguientes para establecer la propiedad:

Constante con nombre	Significado
MQENC_INTEGER_NORMAL	Codificación de entero normal
MQENC_INTEGER_REVERSED	Codificación de entero inverso
MQENC_DECIMAL_NORMAL	Codificación de decimal empaquetado normal
MQENC_DECIMAL_REVERSED	Codificación de decimal empaquetado inverso
MQENC_FLOAT_IEEE_NORMAL	Codificación de coma flotante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codificación de coma flotante IEEE inversa
MQENC_FLOAT_S390	Codificación de coma flotante de arquitectura z/OS
MQENC_NATIVE	Codificación de máquina nativa

Para formar un valor para la propiedad, la aplicación puede añadir tres de estas constantes del modo siguiente:

- Una constante cuyo nombre empieza con MQENC_INTEGER, para especificar la representación de enteros binarios
- Una constante cuyo nombre empieza con MQENC_DECIMAL, para especificar la representación de enteros decimales empaquetados
- Una constante cuyo nombre empieza con MQENC_FLOAT, para especificar la representación de números de coma flotante

De forma alternativa, la aplicación puede establecer la propiedad en MQENC_NATIVE, cuyo valor depende del entorno.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_EXCEPTIONMESSAGE

Tipo de datos:
Cadena

Propiedad de:
Mensaje

Texto que describe la razón por la cual el mensaje se ha enviado al destino de excepciones. Esta propiedad es de solo lectura.

Esta propiedad solo es relevante cuando una aplicación se conecta a un bus de integración de servicios y recibe un mensaje de un destino de excepciones.

JMS_IBM_EXCEPTIONPROBLEMDESTINATION

Tipo de datos:
Cadena

Propiedad de:
Mensaje

El nombre del destino en el que estaba el mensaje antes de que se enviara al destino de excepciones.

Esta propiedad solo es relevante cuando una aplicación se conecta a un bus de integración de servicios y recibe un mensaje de un destino de excepciones.

JMS_IBM_EXCEPTIONREASON

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Un código de razón que indica la razón por la cual el mensaje se envió al destino de excepciones.

Para obtener una lista de todos los códigos de razón posibles, consulte la definición de la clase com.ibm.websphere.sib.SIRCConstants en la documentación generada por la herramienta Javadoc, tal como se proporciona con WebSphere Application Server.

Esta propiedad solo es relevante cuando una aplicación se conecta a un bus de integración de servicios y recibe un mensaje de un destino de excepciones.

JMS_IBM_EXCEPTIONTIMESTAMP

Tipo de datos:
System.Int64

Propiedad de:
Mensaje

La hora cuando se envió el mensaje al destino de excepciones.

La hora se expresa en milisegundos desde 00:00:00 GMT del 1 de enero de 1970.

Esta propiedad solo es relevante cuando una aplicación se conecta a un bus de integración de servicios y recibe un mensaje de un destino de excepciones.

JMS_IBM_FEEDBACK

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Un código que indica la naturaleza de un mensaje de informe.

Los valores válidos de la propiedad son los códigos de retorno y los códigos de razón que se pueden especificar en el campo **Feedback** de un descriptor de mensaje. Para obtener más información sobre el campo **Feedback** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

JMS_IBM_FORMAT

Tipo de datos:
Cadena

Propiedad de:
Mensaje

La naturaleza de los datos de aplicación del mensaje.

Los valores válidos de la propiedad son los mismos que los valores que se pueden especificar en el campo **Format** de un descriptor de mensaje. Para obtener más información sobre el campo **Format** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_LAST_MSG_IN_GROUP

Tipo de datos:
System.Boolean

Propiedad de:
Mensaje

Indica si el mensaje es el último mensaje de un grupo de mensajes.

Establezca la propiedad en true si el mensaje es el último mensaje de un grupo de mensajes. De lo contrario, establezca la propiedad en false, o no establezca la propiedad. De forma predeterminada, la propiedad no está establecida.

El valor true corresponde al distintivo de estado MQMF_LAST_MSG_IN_GROUP, que se puede especificar en el campo **MsgFlags** de un descriptor de mensaje. Para obtener más información sobre este distintivo, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

Esta propiedad se ignora en el dominio Publicar/Suscribir y no es relevante cuando una aplicación se conecta a un bus de integración de servicios.

JMS_IBM_MSGTYPE

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

El tipo del mensaje.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQMT_DATAGRAM	El mensaje es uno que no requiere una respuesta.
MQMT_REQUEST	El mensaje es uno que requiere una respuesta.
MQMT_REPLY	El mensaje es un mensaje de respuesta.
MQMT_REPORT	El mensaje e sun mensaje de informe.

Estos valores corresponden a los tipos de mensaje que se pueden especificar en el campo **MsgType** de un descriptor de mensaje. Para obtener más información sobre el campo **MsgType** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_PUTAPPLTYPE

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

El tipo de la aplicación que envió el mensaje.

Los valores válidos de la propiedad son los tipos de aplicación que se pueden especificar en el campo **PutApplType** de un descriptor de mensaje. Para obtener más información sobre el campo **PutApplType** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_PUTDATE

Tipo de datos:
Cadena

Propiedad de:
Mensaje

La fecha cuando se envió el mensaje.

Los valores válidos de la propiedad son los mismos que los valores que se pueden especificar en el campo **PutDate** de un descriptor de mensaje. Para obtener más información sobre el campo **PutDate** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_PUTTIME

Tipo de datos:
Cadena

Propiedad de:
Mensaje

La hora cuando se envió el mensaje.

Los valores válidos de la propiedad son los mismos que los valores que se pueden especificar en el campo **PutTime** de un descriptor de mensaje. Para obtener más información sobre el campo **PutTime** , consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es relevante cuando la aplicación se conecta a un bus de integración de servicios.

JMS_IBM_REPORT_COA

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar los mensajes de informe 'confirmar en llegada', especificando cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_COA	Solicitar mensajes de informe 'confirmar en llegada', si datos de aplicación del mensaje original incluidos en un mensaje de informe.
MQRO_COA_WITH_DATA	Solicitar mensajes de informe 'confirmar en llegada', con los primeros 100 bytes de datos de aplicación del mensaje original incluidos en un mensaje de informe.
MQRO_COA_WITH_FULL_DATA	Solicitar mensajes de informe 'confirmar en llegada', con todos los datos de aplicación del mensaje original incluidos en un mensaje de informe.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

JMS_IBM_REPORT_COD

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar mensaje de informe 'confirmar en llegada', que especifica cuántos datos d e aplicación del mensaje original se deben incluir en un mensaje de informe.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_COD	Solicitar mensajes de informe 'confirmar en llegada', con ninguno de los datos de aplicación del mensaje original incluido en un mensaje de informe.
MQRO_COD_WITH_DATA	Solicitar mensajes de informe 'confirmar en llegada', con los primeros 100 bytes de datos de aplicación del mensaje original incluidos en un mensaje de informe.
MQRO_COD_WITH_FULL_DATA	Solicitar mensajes de informe 'confirmar en llegada', con todos los datos de aplicación del mensaje original incluidos en un mensaje de informe.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

JMS_IBM_REPORT_DISCARD_MSG

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar que el mensaje se descarte si no se puede entregar a su destino previsto.

Establezca la propiedad en MQRO_DISCARD_MSG para solicitar que el mensaje se descarte si no se puede entregar a su destino previsto. Si necesita que el mensaje se coloque en una cola de mensajes no

entregados en su lugar, o que se envíe a un destino de excepciones, no establezca la propiedad. De forma predeterminada, la propiedad no está establecida.

El valor MQRO_DISCARD_MSG corresponde a una opción de informe que se puede especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre esta opción, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

JMS_IBM_REPORT_EXCEPTION

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar mensajes de informe de excepción, que especifican cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_EXCEPTION	Solicitar mensajes de informe de excepción, sin ningún dato de aplicación del mensajeoriginal incluido en un mensaje de informe.
MQRO_EXCEPTION_WITH_DATA	Solicitar mensajes de informe de excepción, con los primeros 100 bytes de datos de aplicación del mensaje original incluidos en un mensaje de informe.
MQRO_EXCEPTION_WITH_FULL_DATA	Solicitar mensajes de informe de excepción, con todos los datos de aplicación del mensaje original incluidos en un mensaje de informe.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

JMS_IBM_REPORT_EXPIRATION

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar mensajes de informe de caducidad, que especifican cuántos datos de aplicación del mensaje original se deben incluir en un mensaje de informe.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_EXPIRATION	Solicitar mensajes de informe de caducidad, sin ninguno de los datos de aplicación del mensaje original incluido en un mensaje de informe.
MQRO_EXPIRATION_WITH_DATA	Solicitar mensajes de informe de caducidad, con los primeros 100 bytes de datos de aplicación del mensaje original incluidos en un mensaje de informe.

Valor válido	Significado
MQRO_EXPIRATION_WITH_FULL_DATA	Solicitar mensajes de informe de caducidad, con todos los datos de aplicación del mensaje original incluidos en un mensaje de informe.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

De forma predeterminada, la propiedad no está establecida.

JMS_IBM_REPORT_NAN

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar mensajes de informe de notificación de acción negativa.

Establezca la propiedad en MQRO_NAN para solicitar mensajes de informe de notificación de acción negativa. Si no necesita mensajes de informe de notificación de acción negativa, no establezca la propiedad. De forma predeterminada, la propiedad no está establecida.

El valor MQRO_NAN corresponde a una opción de informe que se puede especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre esta opción, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

JMS_IBM_REPORT_PAN

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicitar mensajes de informe de notificación de acción positiva.

Establezca la propiedad en MQRO_PAN para solicitar mensajes de informe de notificación de acción positiva. Si no necesita mensajes de informe de notificación de acción positiva, no establezca la propiedad. De forma predeterminada, la propiedad no está establecida.

El valor MQRO_PAN corresponde a una opción de informe que se puede especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre esta opción, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

JMS_IBM_REPORT_PASS_CORREL_ID

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicite que el identificador de correlación de cualquier mensaje de informe o de respuesta sea el mismo que el identificador de correlación del mensaje original.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_PASS_CORREL_ID	Solicite que el identificador de correlación de cualquier mensaje de informe o de respuesta sea el mismo que el identificador de correlación del mensaje original.
MQRO_COPY_MSG_ID_TO_CORREL_ID	Solicite que el identificador de correlación de cualquier informe o mensaje de respuesta sea el mismo que el identificador de mensaje del mensaje original.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

El valor predeterminado de la propiedad es MQRO_COPY_MSG_ID_TO_CORREL_ID.

JMS_IBM_REPORT_PASS_MSG_ID

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Solicite que el identificador de mensaje de cualquier mensaje de informe o respuesta sea el mismo que el identificador de mensaje del mensaje original.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
MQRO_PASS_MSG_ID	Solicite que el identificador de mensaje de cualquier mensaje de informe o respuesta sea el mismo que el identificador de mensaje del mensaje original.
MQRO_NEW_MSG_ID	Solicite que se genere un nuevo identificador de mensaje para cada mensaje de informe o respuesta.

Estos valores corresponden a opciones de informe que se pueden especificar en el campo **Report** de un descriptor de mensaje. Para obtener más información sobre estas opciones, consulte la publicación *IBM WebSphere MQ Application Programming Reference*.

El valor predeterminado de la propiedad es MQRO_NEW_MSG_ID.

JMS_IBM_RETAIN

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

Establecer esta propiedad indica al gestor de colas que trate un mensaje como una Publicación retenida. Cuando un suscriptor recibe mensajes de temas, podría recibir mensajes adicionales inmediatamente después de la suscripción, más allá de los mensajes recibidos en releases anteriores. Estos mensajes son las publicaciones retenidas opcionales para los temas suscritos. Para cada tema que coincida con la suscripción, si hay una publicación retenida, la publicación pasa a estar disponible para la entrega al consumidor de mensajes de la suscripción.

RETAIN_PUBLICATION es el único valor válido para esta propiedad. De forma predeterminada, esta propiedad no está establecida.

Nota: Esta propiedad solo es relevante en el dominio de publicación/suscripción únicamente.

JMS_IBM_SYSTEM_MESSAGEID

Tipo de datos:
Cadena

Propiedad de:
Mensaje

Un identificador que identifica el mensaje de forma exclusiva dentro del bus de integración de servicios. Esta propiedad es de solo lectura.

Esta propiedad solo es relevante cuando una aplicación se conecta a un bus de integración de servicios.

JMSX_APPID

Tipo de datos:
Cadena

Propiedad de:
Mensaje

El nombre de la aplicación que envió el mensaje.

Esta propiedad es la propiedad definida de JMS con el nombre JMS JMSXAppID. Para obtener más información sobre la propiedad, consulte *Java Message Service Specification, Version 1.1*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

JMSX_DELIVERY_COUNT

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

El número de intentos para entregar el mensaje.

Esta propiedad es la propiedad definida de JMS con el nombre de JMS JMSXDeliveryCount. Para obtener más información sobre la propiedad, consulte *Java Message Service Specification, Version 1.1*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

JMSX_GROUPID

Tipo de datos:
Cadena

Propiedad de:
Mensaje

El identificador del grupo de mensajes al que pertenece el mensaje.

Esta propiedad es la propiedad definida de JMS con el nombre de JMS JMSXGroupID. Para obtener más información sobre la propiedad, consulte *Java Message Service Specification, Version 1.1*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

JMSX_GROUPSEQ

Tipo de datos:
System.Int32

Propiedad de:
Mensaje

El número de secuencia del mensaje de un grupo de mensajes.

Esta propiedad es la propiedad definida de JMS con el nombre de JMS JMSXGroupSeq. Para obtener más información sobre la propiedad, consulte *Java Message Service Specification, Version 1.1*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

JMSX_USERID

Tipo de datos:
Cadena

Propiedad de:
Mensaje

El identificador de usuario asociado con la aplicación que envió el mensaje.

Esta propiedad es la propiedad definida de JMS con el nombre de JMS JMSXUserID. Para obtener más información sobre la propiedad, consulte *Java Message Service Specification, Version 1.1*.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad no es válida para una conexión en tiempo con un intermediario.

XMSC_ASYNC_EXCEPTIONS

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

Esta propiedad determina si XMS informa de un ExceptionListener solo cuando se interrumpe una conexión, o cuando se produce cualquier excepción de forma asíncrona en una llamada de la API XMS. Esta propiedad se aplica a todas las conexiones creadas a partir de esta ConnectionFactory que tienen un ExceptionListener registrado.

Los valores válidos para esta propiedad son:

XMSC_ASYNC_EXCEPTIONS_ALL
Cualquier excepción detectada de forma asíncrona, fuera del ámbito de una llamada a API síncrona, y todas las excepciones de interrupción de conexión se envían al ExceptionListener.

XMSC_ASYNC_EXCEPTIONS_CONNECTIONBROKEN
Solo las excepciones que indican una conexión interrumpida se envían al ExceptionListener. Cualquier otra excepción que se produce durante el proceso asíncrono no se notifica al ExceptionListener y, por lo tanto, la aplicación no está informada de estas excepciones.

De forma predeterminada, esta propiedad está establecida en XMSC_ASYNC_EXCEPTIONS_ALL.

XMSC_CLIENT_ID

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El identificador de cliente para una conexión.

Un identificador de cliente solo se utiliza para dar soporte a suscripciones duraderas en el dominio Publicar/Suscribir y se ignora en el dominio punto a punto. Si desea más información sobre cómo establecer identificadores de cliente, consulte “Objetos ConnectionFactories y Connection” en la página 24.

Esta propiedad no es relevante para una conexión en tiempo real con un intermediario.

XMSC_CONNECTION_TYPE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El tipo del servidor de mensajería al que se conecta una aplicación.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_CT_RTT	Una conexión en tiempo real con un intermediario.
XMSC_CT_WMQ	Una conexión con un gestor de colas IBM WebSphere MQ.
XMSC_CT_WPM	Una conexión a un bus de integración de servicios de WebSphere .

De forma predeterminada, la propiedad no está establecida.

XMSC_DELIVERY_MODE

Tipo de datos:
System.Int32

Propiedad de:
Destino

Nombre utilizado en un URI:

`persistence` (para un destino IBM WebSphere MQ)
`deliveryMode` (para un destino de proveedor de mensajería WebSphere predeterminado)

La modalidad de entrega de mensajes enviados al destino.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_DELIVERY_NOT_PERSISTENT	Un mensaje enviado al destino es no persistente. Se ignora la modalidad de entrega predeterminada del productor de mensajes, o cualquier modalidad de entrega especificada en la llamada Send. Si el destino es una cola IBM WebSphere MQ, el valor del atributo de cola *DefPersistence* también se ignora.
XMSC_DELIVERY_PERSISTENT	Un mensaje enviado al destino es persistente. Se ignora la modalidad de entrega predeterminada del productor de mensajes, o cualquier modalidad de entrega especificada en la llamada Send. Si el destino es una cola IBM WebSphere MQ, el valor del atributo de cola *DefPersistence* también se ignora.

Valor válido	Significado
XMSC_DELIVERY_AS_APP	Un mensaje enviado al destino tiene la modalidad de entrega especificada en la llamada Send. Si la llamada Send no especifica ninguna modalidad de entrega, en su lugar, se utiliza la modalidad de entrega predeterminada del productor de mensajes. Si el destino es una cola IBM WebSphere MQ, se ignora el valor del atributo de cola *DefPersistence*.
XMSC_DELIVERY_AS_DEST	Si el destino es una cola IBM WebSphere MQ, un mensaje colocado en la cola tiene la modalidad de entrega especificada mediante el valor del atributo de cola *DefPersistence*. Se ignora la modalidad de entrega predeterminada del productor de mensajes, o cualquier modalidad de entrega especificada en la llamada Send. Si el destino no es una cola IBM WebSphere MQ, el significado es el mismo que para XMSC_DELIVERY_AS_APP.

El valor predeterminado es XMSC_DELIVERY_AS_APP.

XMSC_IC_PROVIDER_URL

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Se utiliza para localizar el directorio de denominación JNDI de forma que no es necesario que el servicio de denominación COS esté en el mismo servidor que el servicio web.

XMSC_IC_SECURITY_AUTHENTICATION

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Basado en la interfaz de contexto Java SECURITY_AUTHENTICATION. Esta propiedad solo es aplicable al contexto de denominación COS.

XMSC_IC_SECURITY_CREDENTIALS

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Basado en la interfaz de contexto Java SECURITY_CREDENTIALS. Esta propiedad solo es aplicable al contexto de denominación COS.

XMSC_IC_SECURITY_PRINCIPAL

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Basado en la interfaz de contexto Java SECURITY_PRINCIPAL. Esta propiedad solo es aplicable al contexto de denominación COS.

XMSC_IC_SECURITY_PROTOCOL

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Basado en la interfaz de contexto Java SECURITY_PROTOCOL Esta propiedad sólo es aplicable al contexto de denominación COS.

XMSC_IC_URL

Tipo de datos:
Cadena

Propiedad de:
InitialContext

Para contextos de LDAP y FileSystem, la dirección del repositorio que contiene objetos administrados.

Para contextos de denominación COS, la dirección del servicio web que busca los objetos en el directorio.

XMSC_IS_SUBSCRIPTION_MULTICAST

Tipo de datos:
System.Boolean

Propiedad de:
MessageConsumer

Indica si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport. Esta propiedad es de solo lectura.

El valor de la propiedad es true si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport. De lo contrario, el valor es false.

Esta propiedad solo es relevante para una conexión en tiempo real con un intermediario.

XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST

Tipo de datos:
System.Boolean

Propiedad de:
MessageConsumer

Indica si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport con una calidad de servicio fiable. Esta propiedad es de solo lectura.

El valor de la propiedad es true si los mensajes se van a entregar al consumidor de mensajes mediante WebSphere MQ Multicast Transport con una calidad de servicio fiable. De lo contrario, el valor es false.

Esta propiedad solo es relevante para una conexión en tiempo real con un intermediario.

XMSC_JMS_MAJOR_VERSION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionMetaData

El número de versión principal de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.

XMSC_JMS_MINOR_VERSION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionMetaData

El número de versión menor de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.

XMSC_JMS_VERSION

Tipo de datos:
Cadena

Propiedad de:
ConnectionMetaData

El identificador de versión de la especificación JMS en la que se basa XMS . Esta propiedad es de solo lectura.

XMSC_MAJOR_VERSION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionMetaData

El número de versión del cliente XMS. Esta propiedad es de solo lectura.

XMSC_MINOR_VERSION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionMetaData

El número de release del cliente XMS. Esta propiedad es de solo lectura.

XMSC_PASSWORD

Tipo de datos:
Matriz de bytes

Propiedad de:
ConnectionFactory

Una contraseña que se puede utilizar para autenticar la aplicación cuando intenta conectarse a un servidor de mensajería. La contraseña se utiliza con la propiedad XMSC_USERID.

De forma predeterminada, la propiedad no está establecida.

Si se conecta a plataformas distribuidas de IBM WebSphere MQ y establece la propiedad XMSC_USERID de la fábrica de conexiones, su valor debe coincidir con el valor **userid** del usuario conectado. Si no establece estas propiedades, el gestor de colas utiliza el **userid** del usuario conectado de forma predeterminada. Si necesita una autenticación de nivel de conexión adicional de usuarios individuales, puede escribir una salida de autenticación de cliente, que se configura en IBM WebSphere MQ. Puede obtener más información sobre cómo crear una salida de autenticación de cliente en el tema Autenticación del manual de Clientes de IBM WebSphere MQ.

Para autenticar el usuario al conectarse a IBM WebSphere MQ en z/OS debe utilizar una salida de seguridad.

XMSC_PRIORITY

Tipo de datos:
System.Int32

Propiedad de:
Destino

Nombre utilizado en un URI:
priority

La prioridad de los mensajes enviados al destino.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
Un entero dentro del rango de 0, la prioridad más baja, a 9, la prioridad más alta.	Un mensaje enviado al destino tiene la prioridad especificada. La prioridad predeterminada del productor de mensajes, o cualquier prioridad especificada en la llamada Send, se ignora. Si el destino es una cola IBM WebSphere MQ, también se ignora el valor del atributo de cola **DefPriority**.
XMSC_PRIORITY_AS_APP	Un mensaje enviado al destino tiene la prioridad especificada en la llamada Send. Si la llamada Send no especifica ninguna prioridad, en su lugar, se utiliza la prioridad predeterminada del productor de mensajes. Si el destino es una cola IBM WebSphere MQ, se ignora el valor del atributo de cola **DefPriority**.
XMSC_PRIORITY_AS_DEST	Si el destino es una cola IBM WebSphere MQ, un mensaje colocado en la cola tiene la prioridad especificada por el valor del atributo de cola **DefPriority**. La prioridad predeterminada del productor de mensajes, o cualquier prioridad especificada en la llamada Send, se ignora. Si el destino no es una cola IBM WebSphere MQ, el significado es el mismo que el de XMSC_PRIORITY_AS_APP.

El valor predeterminado es XMSC_PRIORITY_AS_APP.

WebSphere MQ Real-Time Transport y WebSphere MQ Multicast Transport no realizan ninguna acción basándose en la prioridad de un mensaje.

XMSC_PROVIDER_NAME

Tipo de datos:
Cadena

Propiedad de:
ConnectionMetaData

El proveedor del cliente XMS. Esta propiedad es de solo lectura.

XMSC_RTT_BROKER_PING_INTERVAL

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El intervalo de tiempo, en milisegundos, transcurrido el cual XMS .NET comprueba la conexión con un servidor de mensajería en tiempo real para detectar cualquier actividad. Si no se detecta ninguna

actividad, el cliente inicia una conexión ping; la conexión se cierra si no se detecta ninguna respuesta de ping.

El valor predeterminado de la propiedad es 30000.

XMSC_RTT_CONNECTION_PROTOCOL

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El protocolo de comunicación utilizado para una conexión en tiempo real con un intermediario.

El valor de la propiedad debe ser XMSC_RTT_CP_TCP, lo que significa un conexión en tiempo real con un intermediario sobre TCP/IP. El valor predeterminado es XMSC_RTT_CP_TCP.

XMSC_RTT_HOST_NAME

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de host o la dirección IP del sistema en el que se ejecuta un intermediario.

Esta propiedad se utiliza con la propiedad XMSC_RTT_PORT para identificar el intermediario.

De forma predeterminada, la propiedad no está establecida.

XMSC_RTT_LOCAL_ADDRESS

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de host o la dirección IP de la interfaz de red local que se va a utilizar para una conexión en tiempo real con un intermediario.

Esta propiedad solo es útil si el sistema en el cual se está ejecutando la aplicación tiene dos o más interfaces de red y necesita poder especificar qué interfaz se debe utilizar para una conexión en tiempo real. Si el sistema tiene solo una interfaz de red, solo se puede utilizar dicha interfaz. Si el sistema tiene dos o más interfaces de red y la propiedad no está establecida, la interfaz se selecciona de forma aleatoria.

De forma predeterminada, la propiedad no está establecida.

XMSC_RTT_MULTICAST

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory y Destination

Nombre utilizado en un URI:
mulicast

El valor de multidifusión para una fábrica de conexiones o destino. Solo un destino que es un tema puede tener esta propiedad.

Una aplicación utiliza esta propiedad para habilitar la multidifusión con una conexión en tiempo real con un intermediario y, si la multidifusión está habilitada, para especificar la forma precisa en la cual se utiliza

la multidifusión para entregar mensajes del intermediario a un consumidor de mensajes. La propiedad no tiene ningún efecto sobre cómo un productor de mensajes envía mensajes al intermediario.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_RTT_MULTICAST_DISABLED	Los mensajes no se entregan a un consumidor de mensajes utilizando WebSphere MQ Multicast Transport. Este valor es el valor predeterminado para un objeto ConnectionFactory.
XMSC_RTT_MULTICAST_ASCF	Los mensajes se entregan a un consumidor de mensajes de acuerdo con el valor de multidifusión para la fábrica de conexiones asociada al consumidor de mensajes. El valor de multidifusión para la fábrica de conexiones se indica en el momento cuando se crea la conexión. Este valor solo es válido para un objeto Destination, y es el valor predeterminado para un objeto Destination.
XMSC_RTT_MULTICAST_ENABLED	Si el tema se configura para la multidifusión en el intermediario, los mensajes se entregan a un consumidor de mensajes utilizando WebSphere MQ Multicast Transport. Se utiliza una calidad de servicio fiable si el tema está configurado para la multidifusión fiable.
XMSC_RTT_MULTICAST_RELIABLE	Si el tema se ha configurado para la multidifusión fiable en el intermediario, los mensajes se entregan a un consumidor de mensajes utilizando WebSphere MQ Multicast Transport con una calidad de servicio fiable. Si el tema no está configurado para la multidifusión fiable, no puede crear un consumidor de mensajes para el tema.
XMSC_RTT_MULTICAST_NOT_RELIABLE	Si el tema se configura para la multidifusión en el intermediario, los mensajes se entregan a un consumidor de mensajes utilizando WebSphere MQ Multicast Transport. No se utiliza una calidad de servicio fiable, aunque el tema se haya configurado para la multidifusión fiable.

XMSC_RTT_PORT

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El número del puerto en el cual un intermediario escucha solicitudes entrantes. En el intermediario, debe configurar el nodo de proceso de mensajes Real-timeInput o Real-timeOptimizedFlow para escuchar en este puerto.

Esta propiedad se utiliza con la propiedad XMSC_RTT_HOST_NAME para identificar el intermediario.

El valor predeterminado de la propiedad es XMSC_RTT_DEFAULT_PORT, o 1506.

XMSC_TIME_TO_LIVE

Tipo de datos:
System.Int32

Propiedad de:
Destino

Nombre utilizado en un URI:

expiry (para un destino IBM WebSphere MQ)
timeToLive (para un destino de proveedor de mensajería predeterminado de WebSphere)

El tiempo de vida para los mensajes enviados al destino.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
0	Un mensaje enviado al destino nunca caduca.
Un entero positivo	Un mensaje enviado al destino tiene el tiempo de vida especificado en milisegundos. Se ignora el periodo de vida predeterminado del productor de mensajes, o cualquier periodo de vida especificado en la llamada Send.
XMSC_TIME_TO_LIVE_AS_APP	Un mensaje enviado al destino tiene el periodo de tiempo especificado en la llamada Send. Si la llamada Send no especifica ningún periodo de tiempo, en su lugar, se utiliza el periodo de tiempo predeterminado del productor de mensajes.

El valor predeterminado es XMSC_TIME_TO_LIVE_AS_APP.

XMSC_USERID

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Un identificador de usuario que se puede utilizar para autenticar la aplicación cuando intenta conectarse a un servidor de mensajería. El identificador de usuario se utiliza con la propiedad XMSC_PASSWORD.

De forma predeterminada, la propiedad no está establecida.

Si se conecta a plataformas distribuidas de IBM WebSphere MQ y establece la propiedad XMSC_USERID de la fábrica de conexiones, su valor debe coincidir con el valor **userid** del usuario conectado. Si no establece estas propiedades, el gestor de colas utiliza el **userid** del usuario conectado de forma predeterminada. Si necesita más autenticación a nivel de conexión de usuarios individuales, puede escribir una salida de autenticación de cliente que esté configurada en IBM WebSphere MQ.

Para autenticar el usuario al conectarse a IBM WebSphere MQ en z/OS debe utilizar una salida de seguridad.

XMSC_VERSION

Tipo de datos:
Cadena

Propiedad de:
ConnectionMetaData

El identificador de versión del cliente XMS. Esta propiedad es de solo lectura.

XMSC_WMQ_BROKER_CONTROLQ

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre del gestor de colas utilizado por un intermediario.

El valor predeterminado de la propiedad es SYSTEM.BROKER.CONTROL.QUEUE.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WMQ_BROKER_PUBQ

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de la cola supervisada por un intermediario donde las aplicaciones envían mensajes que publican.

El valor predeterminado de la propiedad es SYSTEM.BROKER.DEFAULT.STREAM.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WMQ_BROKER_QMGR

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre del gestor de colas al que está conectado un intermediario.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WMQ_BROKER_SUBQ

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de la cola de suscriptor para un consumidor de mensajes no duraderos.

El nombre de la cola de suscriptor debe empezar con los caracteres siguientes:

SYSTEM.JMS.ND.

Si desea que todos los consumidores de mensajes no duraderos compartan una cola de suscriptor, especifique el nombre completo de la cola compartida. Debe existir una cola con el nombre especificado antes de que una aplicación pueda crear un consumidor de mensajes no duraderos.

Si desea que cada consumidor de mensajes no duraderos recupere mensajes de su propia cola de suscriptores exclusiva, especifique un nombre de cola que termine con un asterisco (*). A continuación, cuando una aplicación crea un consumidor de mensajes no duradero, el cliente de XMS crea una cola dinámica para uso exclusivo del consumidor de mensajes. El cliente XMS utiliza el valor de la propiedad para establecer el contenido del campo **DynamicQName** en el descriptor de objeto que se utiliza para crear la cola dinámica.

El valor predeterminado de la propiedad es SYSTEM.JMS.ND.SUBSCRIBER.QUEUE, lo que significa que XMS utiliza el enfoque de cola compartida de forma predeterminada.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WMQ_BROKER_VERSION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory y Destination

Nombre utilizado en un URI:
brokerVersion

El tipo de intermediario utilizado por la aplicación para una conexión o para el destino. Solo un destino que es un tema puede tener esta propiedad.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WMQ_BROKER_V1	La aplicación está utilizando un intermediario IBM WebSphere MQ Publicar/Suscribir. La aplicación también puede utilizar este valor si migra de IBM WebSphere MQ Publicar/Suscribir a WebSphere Event Broker o WebSphere Message Broker pero no ha cambiado la aplicación.
XMSC_WMQ_BROKER_V2	La aplicación utiliza un intermediario de WebSphere Event Broker o WebSphere Message Broker.
XMSC_WMQ_BROKER_UNSPECIFIED	Después de migrar el intermediario de la versión 6 a la versión 7, establezca esta propiedad para que las cabeceras RFH2 ya no se utilicen. Después de la migración, esta propiedad deja de ser relevante.

El valor predeterminado para una fábrica de conexiones es XMSC_WMQ_BROKER_UNSPECIFIED pero, de forma predeterminada, la propiedad no está establecida para un destino. Establecer la propiedad para un destino sustituirá cualquier valor especificado por la propiedad de la fábrica de conexiones.

XMSC_WMQ_CCDTURL

Tipo de datos:
System.String

Propiedad de:
ConnectionFactory

Un localizador uniforme de recursos (URL) que identifica el nombre y la ubicación del archivo que contiene la tabla de la definición de canal de cliente y, también, especifica cómo se puede acceder al archivo.

De forma predeterminada, esta propiedad no está establecida.

XMSC_WMQ_CCSID

Tipo de datos:
System.Int32

Propiedad de:
Destino

Nombre utilizado en un URI:
CCSID

El identificador (CCSID) del juego de caracteres codificado, o página de códigos, en el que están escritas las series de datos de carácter en el cuerpo de un mensaje cuando el cliente de XMS envía el mensaje al destino. Si se establece para un mensaje individual, la propiedad JMS_IBM_CHARACTER_SET altera temporalmente el CCSID especificado para el destino con esta propiedad.

El valor predeterminado de la propiedad es 1208.

Esta propiedad es relevante solo para los mensajes enviados al destino, no para los mensajes recibidos del destino.

XMSC_WMQ_CHANNEL

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre del canal que se va a utilizar para una conexión.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente.

XMSC_WMQ_CLIENT_RECONNECT_OPTIONS

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Esta propiedad especifica las opciones de reconexión de cliente para las conexiones nuevas creadas por esta fábrica de conexiones. Se encuentra en XMSC, y es una de:

- WMQ_CLIENT_RECONNECT_AS_DEF (default). Utilice el valor especificado en el archivo `mqclient.ini`. Establezca el valor utilizando la propiedad **DefRecon** dentro de la stanza Channels. Se puede establecer en uno de estos valores:
  1. Sí. Se comporta como la opción WMQ_CLIENT_RECONNECT
  2. NO. Valor por omisión. No especifica ninguna opción de reconexión.
  3. QMGR. Se comporta como la opción WMQ_CLIENT_RECONNECT_Q_MGR
  4. DISABLED. Se comporta como la opción WMQ_CLIENT_RECONNECT_DISABLED
- WMQ_CLIENT_RECONNECT. Reconectarse a cualquiera de los gestores de colas especificados en la lista de nombres de conexión.
- WMQ_CLIENT_RECONNECT_Q_MGR. Se reconecta al mismo gestor de colas al que estaba conectado originalmente. Devuelve `MQRC_RECONNECT_QMID_MISMATCH` si el gestor de colas al que intenta conectarse (especificado en la lista de nombres de conexión) tiene un QMID diferente al del gestor de colas al que estaba conectado originalmente.
- WMQ_CLIENT_RECONNECT_DISABLED. La reconexón está inhabilitada.

XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Esta propiedad especifica la duración de tiempo, en segundos, que una conexión de cliente intenta reconectarse.

Después de intentar reconectarse para esta duración de tiempo, el cliente fallará con MQRC_RECONNECT_FAILED. El valor predeterminado para esta propiedad es XMSC.WMQ_CLIENT_RECONNECT_TIMEOUT_DEFAULT.

El valor predeterminado de esta propiedad es 1800.

XMSC_WMQ_CONNECTION_MODE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

La modalidad a través de la cual una aplicación se conecta a un gestor de colas.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WMQ_CM_BINDINGS	Una conexión con un gestor de colas en modalidad de enlaces, para un rendimiento óptimo. Este valor es el valor predeterminado para C/C++.
XMSC_WMQ_CM_CLIENT	Una conexión con un gestor de colas en modalidad cliente, para garantizar una pila totalmente gestionada. Este valor es el valor predeterminado para .NET.
XMSC_WMQ_CM_CLIENT_UNMANAGED (solo para .NET)	Una conexión con un gestor de colas que fuerza una pila de cliente no gestionada.

Conceptos relacionados

Operaciones gestionadas o no gestionadas en .NET
El código gestionado se ejecuta de forma exclusiva dentro del entorno de tiempo de ejecución de lenguaje común de .NET y depende por completo de los servicios proporcionados por ese tiempo de ejecución. Una aplicación se clasifica como no gestionada si alguna parte de la aplicación se ejecuta o llama a servicios fuera del entorno de tiempo de ejecución de lenguaje común de .NET.

XMSC_WMQ_CONNECTION_NAME_LIST

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Esta propiedad especifica los hosts a los que el cliente intenta reconectarse después de que se hayan interrumpido sus conexiones.

La lista de nombres de conexión es una lista separada por comas de pares de host/puerto ip. El valor predeterminado para esta propiedad es WMQ_CONNECTION_NAME_LIST_DEFAULT.

Por ejemplo, `127.0.0.1(1414),host2.example.com(1400)`

El valor predeterminado de esta propiedad es `localhost (1414)`.

XMSC_WMQ_DUR_SUBQ

Tipo de datos:
Cadena

Propiedad de:
Destino

El nombre de la cola de suscriptor para un suscriptor duradero que está recibiendo mensajes del destino. Solo un destino que es un tema puede tener esta propiedad.

El nombre de la cola de suscriptor debe empezar con los caracteres siguientes:

SYSTEM.JMS.D.

Si desea que todos los suscriptores duraderos compartan una cola de suscriptor, especifique el nombre completo de la cola compartida. Una cola con el nombre especificado debe existir antes de que una aplicación pueda crear un suscriptor duradero.

Si desea que cada suscriptor duradero recupere mensajes de su propia cola de suscriptor exclusiva, especifique un nombre de cola que termine con un asterisco (*). A continuación, cuando una aplicación crea un suscriptor duradero, el cliente XMS crea una cola dinámica para uso exclusivo del suscriptor duradero. El cliente XMS utiliza el valor de la propiedad para establecer el contenido del campo **DynamicQName** en el descriptor de objeto que se utiliza para crear la cola dinámica.

El valor predeterminado de la propiedad es SYSTEM.JMS.D.SUBSCRIBER.QUEUE, lo que significa que XMS utiliza el enfoque de cola compartida de forma predeterminada.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WMQ_ENCODING

Tipo de datos:
System.Int32

Propiedad de:
Destino

Cómo se representan datos numéricos en el cuerpo de un mensaje cuando el cliente XMS envía el mensaje al destino. Si se establece para un mensaje individual, la propiedad JMS_IBM_ENCODING sustituye la codificación especificada para el destino a través de esta propiedad. La propiedad especifica la representación de enteros binarios, enteros decimales empaquetados y números de coma flotante.

Los valores válidos de la propiedad son los mismos que los valores que se pueden especificar en el campo **Encoding** de un descriptor de mensaje.

Una aplicación puede utilizar las constantes con nombre siguientes para establecer la propiedad:

Constante con nombre	Significado
MQENC_INTEGER_NORMAL	Codificación de entero normal
MQENC_INTEGER_REVERSED	Codificación de entero inverso
MQENC_DECIMAL_NORMAL	Codificación de decimal empaquetado normal
MQENC_DECIMAL_REVERSED	Codificación de decimal empaquetado inverso
MQENC_FLOAT_IEEE_NORMAL	Codificación de coma flotante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codificación de coma flotante IEEE inversa
MQENC_FLOAT_S390	Codificación de coma flotante de arquitectura z/OS
MQENC_NATIVE	Codificación de máquina nativa

Para formar un valor para la propiedad, la aplicación puede añadir tres de estas constantes del modo siguiente:

- Una constante cuyo nombre empieza con MQENC_INTEGER, para especificar la representación de enteros binarios
- Una constante cuyo nombre empieza con MQENC_DECIMAL, para especificar la representación de enteros decimales empaquetados
- Una constante cuyo nombre empieza con MQENC_FLOAT, para especificar la representación de números de coma flotante

De forma alternativa, la aplicación puede establecer la propiedad en MQENC_NATIVE, cuyo valor depende del entorno.

El valor predeterminado de la propiedad es MQENC_NATIVE.

Esta propiedad es relevante solo para los mensajes enviados al destino, no para los mensajes recibidos del destino.

XMSC_WMQ_FAIL_IF_QUIESCE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory y Destination

Nombre utilizado en un URI:
failIfQuiesce

Indica si las llamadas a determinados métodos fallan si el gestor de colas al que está conectada la aplicación está en un estado de inmovilización.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WMQ_FIQ_YES	Las llamadas a determinados métodos fallan si el gestor de colas está en un estado de inmovilización. Cuando la aplicación detecta que el gestor de colas está en fase de inmovilización, la aplicación puede completar su tarea inmediata y cerrar la conexión, lo que permite que se detenga el gestor de colas.
XMSC_WMQ_FIQ_NO	No falla ninguna llamada de método porque el gestor está en un estado de inmovilización. Si especifica esta valor, la aplicación no puede detectar que el gestor de colas está en fase de inmovilización. La aplicación podría seguir realizando operaciones en el gestor de colas y, por lo tanto, impedir que se detenga el gestor de colas.

El valor predeterminado para una fábrica de conexiones es XMSC_WMQ_FIQ_YES pero, de forma predeterminada, la propiedad no está establecida para un destino. Establecer la propiedad para un destino sustituirá cualquier valor especificado por la propiedad de la fábrica de conexiones.

XMSC_WMQ_MESSAGE_BODY

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina si una aplicación XMS procesa la MQRFH2 de un mensaje IBM WebSphere MQ como parte de la carga útil del mensaje (es decir, como parte del cuerpo del mensaje).

Nota: Al enviar mensajes a un destino, la propiedad XMSC_WMQ_MESSAGE_BODY sustituye la propiedad de destino XMS existente XMSC_WMQ_TARGET_CLIENT.

Los valores válidos para esta propiedad son:

XMSC_WMQ_MESSAGE_BODY_JMS
Recibir: El tipo de mensaje XMS de entrada y el cuerpo se determinan a través del contenido de la MQRFH2 (si está presente) o MQMD (si no hay ninguna MQRFH2) en el mensaje IBM WebSphere MQ recibido.

Enviar: El cuerpo del mensaje XMS de salida contiene una cabecera MQRFH2 generada automáticamente y añadida como prefijo basándose en las propiedades de mensaje XMS y los campos de cabecera.

XMSC_WMQ_MESSAGE_BODY_MQ
Recibir: El tipo de mensaje XMS de entrada siempre es ByteMessage, independientemente de los contenidos del mensaje IBM WebSphere MQ recibido o el campo de formato del MQMD recibido.

El cuerpo del mensaje XMS está formado por los datos de mensaje sin modificar devueltos por la llamada de la API del proveedor de mensajería subyacente. El juego de caracteres y la codificación de los datos en el cuerpo del mensaje se determina mediante los campos CodedCharSetId y Encoding del MQMD. El formato de los datos del cuerpo del mensaje se determina mediante el campo Format del MQMD.

Enviar: El cuerpo del mensaje XMS de salida contiene la carga útil de la aplicación tal cual; y no se añade ninguna cabecera IBM WebSphere MQ generada automáticamente al cuerpo.

XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED

Recibir: El cliente XMS determina un valor adecuado para esta propiedad. En la vía de acceso de recepción, este valor es el valor de propiedad WMQ_MESSAGE_BODY_JMS.

Enviar: El cliente XMS determina un valor adecuado para esta propiedad. En la vía de acceso de envío, este valor es el valor de propiedad XMSC_WMQ_TARGET_CLIENT.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED.

XMSC_WMQ_MQMD_MESSAGE_CONTEXT

Tipo de datos:

System.Int32

Propiedad de:

Destino

Determina qué nivel de contexto de mensaje debe establecer la aplicación XMS . La aplicación debe estar en ejecución con la autoridad de contexto adecuada para que esta propiedad tenga efecto.

Los valores válidos para esta propiedad son:

XMSC_WMQ_MDCTX_DEFAULT

Para mensajes de salida, la llamada de la API MQOPEN y la estructura MQPMO no especifican ninguna opción de contexto de mensaje explícita.

XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT

La llamada de la API MQOPEN especifica la opción de contexto de mensaje MQOO_SET_IDENTITY_CONTEXT y la estructura MQPMO especificaMQPMO_SET_IDENTITY_CONTEXT.

XMSC_WMQ_MDCTX_SET_ALL_CONTEXT

La llamada de la API MQOPEN especifica la opción de contexto de mensaje MQOO_SET_ALL_CONTEXT y la estructura MQPMO especifica MQPMO_SET_ALL_CONTEXT.

De forma predeterminada, esta propiedad está establecida enXMSC_WMQ_MDCTX_DEFAULT.

Nota: Esta propiedad no es relevante cuando una aplicación se conecta al bus de integración del sistema.

Las propiedades siguientes requieren que la propiedad XMSC_WMQ_MQMD_MESSAGE_CONTEXT esté establecida en el valor de propiedad XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT o el valor de propiedad XMSC_WMQ_MDCTX_SET_ALL_CONTEXT al enviar un mensaje para poder tener el efecto deseado:

- JMS_IBM_MQMD_USERIDENTIFIER
- JMS_IBM_MQMD_ACCOUNTINGTOKEN
- JMS_IBM_MQMD_APPLIDENTITYDATA

Las propiedades siguientes requieren que la propiedad XMSC_WMQ_MQMD_MESSAGE_CONTEXT esté establecida en el valor de propiedad XMSC_WMQ_MDCTX_SET_ALL_CONTEXT al enviar un mensaje para poder tener el efecto deseado.

- JMS_IBM_MQMD_PUTAPPLTYPE
- JMS_IBM_MQMD_PUTAPPLNAME

- JMS_IBM_MQMD_PUTDATE
- JMS_IBM_MQMD_PUTTIME
- JMS_IBM_MQMD_APPLORIGINDATA

XMSC_WMQ_MQMD_READ_ENABLED

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina si una aplicación XMS puede extraer los valores de los campos MQMD o no.

Los valores válidos para esta propiedad son:

XMSC_WMQ_READ_ENABLED_NO
Al enviar mensajes, las propiedades JMS_IBM_MQMD* en un mensaje enviado no se actualizan para reflejar los valores de campo actualizados en el MQMD.

Al recibir mensajes, ninguna de las propiedades JMS_IBM_MQMD* está disponible en un mensaje recibido, aunque el emisor haya establecido algunas o todas las propiedades.

XMSC_WMQ_READ_ENABLED_YES
Al enviar mensajes, todas las propiedades JMS_IBM_MQMD* en un mensaje enviado se actualizan para reflejar los valores de campo actualizados en el MQMD, incluyendo aquellas propiedades que el emisor no ha establecido de forma explícita.

Al recibir mensajes, todas las propiedades JMS_IBM_MQMD* están disponibles en un mensaje recibido, incluyendo aquellas propiedades que el emisor no ha establecido explícitamente.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ_READ_ENABLED_NO.

XMSC_WMQ_MQMD_WRITE_ENABLED

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina si una aplicación XMS puede indicar los valores de los campos MQMD o no.

Los valores válidos para esta propiedad son:

XMSC_WMQ_WRITE_ENABLED_NO
Todas las propiedades JMS_IBM_MQMD* se ignoran y sus valores no se copian en la estructura MQMD subyacente.

XMSC_WMQ_WRITE_ENABLED_YES
Se procesan las propiedades JMS_IBM_MQMD*. Sus valores se copian en la estructura MQMD subyacente.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ_WRITE_ENABLED_NO.

XMSC_WMQ_PUT_ASYNC_ALLOWED

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina si los productores de mensajes están permitidos para utilizar operaciones de transferencia asíncrona para enviar mensajes a este destino.

Los valores válidos para esta propiedad son:

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST
Determine si las operaciones de transferencia asíncrona están permitidas consultando la definición de cola o tema.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_Q_DEF
Determine si las operaciones de transferencia asíncrona están permitidas consultando la definición de cola.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_TOPIC_DEF
Determine si las operaciones de transferencia asíncrona están permitidas consultando la definición de tema.

XMSC_WMQ _PUT_ASYNC_ALLOWED_DISABLED
Las operaciones de transferencia asíncrona no están permitidas.

XMSC_WMQ _PUT_ASYNC_ALLOWED_ENABLED
Las operaciones de transferencia asíncrona están permitidas.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST.

Nota: Esta propiedad no es relevante cuando una aplicación se está conectando al bus de integración del sistema.

XMSC_WMQ_READ_AHEAD_ALLOWED

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina si los consumidores de mensajes y los navegadores de colas están autorizados para utilizar la lectura anticipada para obtener mensajes no persistentes y no transaccionales de este destino en un almacenamiento intermedio interno antes de recibirlos.

Los valores válidos para esta propiedad son:

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF
Determine si se permite la lectura anticipada consultando la definición de cola.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_ TOPIC _DEF
Determine si se permite la lectura anticipada consultando la definición de tema.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_DEST
Determine si se permite la lectura anticipada consultando la definición de cola o tema.

XMSC_WMQ_READ_AHEAD_ALLOWED_DISABLED
La lectura anticipada no está permitida mientras se consumen o navegan mensajes.

XMSC_WMQ_READ_AHEAD_ALLOWED_ENABLED
La lectura anticipada está permitida.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ _READ_AHEAD_ALLOWED_AS_DEST.

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY

Tipo de datos:
System.Int32

Propiedad de:
Destino

Esta propiedad determina, para los mensajes que se están entregando a un escucha de mensajes asíncronos, qué sucede con los mensajes en el almacenamiento de lectura anticipada interno, cuando se cierra el consumidor de mensajes.

Esta propiedad es aplicable en la especificación de opciones de cierre de cola al consumir mensajes de un destino y no es aplicable al enviar mensajes a un destino.

Esta propiedad se ignora para los Examinadores de colas porque, durante la exploración, los mensajes siguen estando disponibles en las colas.

Los valores válidos para esta propiedad son:

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT
Solo se completa la invocación del escucha de mensajes actual antes de la devolución, posiblemente dejando mensajes en el almacenamiento intermedio de lectura anticipada interno que, después, se descartan.

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_ALL
Todos los mensajes del almacenamiento intermedio de lectura anticipada interno se entregan al escucha de mensajes de aplicación antes de la devolución.

De forma predeterminada, esta propiedad está establecida en XMSC_WMQ _READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT.

Nota:

- **Terminación anómala de aplicación**

  Todos los mensajes del almacenamiento intermedio de lectura anticipada se pierden cuando una aplicación XMS termina de forma abrupta.

- **Implicaciones en transacciones**

  La lectura anticipada está inhabilitada cuando las aplicaciones utilizan transacciones. De esta forma, la aplicación no está viendo ninguna diferencia en el comportamiento cuando utilizando sesiones con transacción.

- **Implicaciones de modalidades de acuse de recibo de sesión**

  La lectura anticipada está habilitada en una sesión no transaccional cuando las modalidades de acuse de recibo son XMSC_AUTO_ACKNOWLEDGE o XMSC_DUPS_OK_ACKNOWLEDGE. La lectura anticipada está inhabilitada cuando la modalidad de acuse de recibo de la sesión es XMSC_CLIENT_ACKNOWLEDGE, con independencia de si la sesión es transaccional o no.

- **Implicaciones en los Examinadores de colas y Selectores de examinadores de colas**

  Los Examinadores de colas y los Selectores de examinadores de colas, utilizados en aplicaciones XMS, obtienen la ventaja de rendimiento de la lectura anticipada. El cierre del Examinador de colas no degrada el rendimiento, porque el mensaje sigue estando disponible en la cola para cualquier operación adicional. No existe ninguna otra implicación en los examinadores de colas y los selectores de examinadores de colas aparte de las ventajas de rendimiento de la lectura anticipada.

- **Implicaciones de las propiedades de destino de lectura anticipada en WebSphere Message Broker V6 o gestores de colas anteriores**

  Si especifica las propiedades de destino XMSC_WMQ_READ_AHEAD_ALLOWED y XMSC_WMQ_READ_AHEAD_CLOSE_POLICY, cuando la aplicación XMS utilice el gestor de colas WebSphere Message Broker V6 no podrá utilizar los valores especificados. Estos valores de propiedad de destino se ignorarán de forma silenciosa y las aplicaciones seguirán funcionando sin lectura anticipada. No se emitirá ningún error cuando se utilice con gestores de colas V6 .

XMSC_WMQ_HOST_NAME

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de host o la dirección IP del sistema en el cual se ejecuta un gestor de colas.

Esta propiedad solo se utiliza cuando una aplicación se conecta a un gestor de colas en modalidad cliente. La propiedad se utiliza con la propiedad XMSC_WMQ_PORT para identificar el gestor de colas.

El valor predeterminado de la propiedad es `localhost`.

XMSC_WMQ_LOCAL_ADDRESS

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Para una conexión a un gestor de colas, esta propiedad especifica la interfaz de red local que se va a utilizar, o el puerto local o el rango de puertos locales que se van a utilizar, o ambos.

El valor de la propiedad es una serie con el formato siguiente:

[*nombre_host*][(*puerto_bajo*)[,*puerto_alto*])]

Los significados de las variables son los siguientes:

nombre_host
El nombre de host o la dirección IP de la interfaz de red local que se va a utilizar para la conexión.

Proporcionar esta información solo es necesario si el sistema en el cual se está ejecutando la aplicación tiene dos o más interfaces de red y tendrá que poder especificar qué interfaz se debe utilizar para la conexión. Si el sistema tiene solo una interfaz de red, solo se puede utilizar dicha interfaz. Si el sistema tiene dos o más interfaces de red y no especifica qué interfaz se debe utilizar, la interfaz se selecciona de forma aleatoria.

puerto_bajo
El número del puerto local que se va a utilizar para la conexión.

Si *puerto_alto* también se especifica, *puerto_bajo* se interpreta como el número de puerto inferior en un rango de números de puerto.

puerto_alto
El número de puerto superior en un rango de números de puerto. Se debe utilizar uno de los puertos del rango especificado para la conexión.

La longitud máxima de la serie es 48 caracteres.

Aquí aparecen algunos ejemplos de valores válidos de la propiedad:

JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente.

XMSC_WMQ_MESSAGE_SELECTION

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

Determina si la selección de mensajes ha sido realizada por el cliente de XMS o el intermediario.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WMQ_MSEL_CLIENT	La selección de mensajes ha sido realizada por el cliente de XMS.
XMSC_WMQ_MSEL_BROKER	La selección de mensajes ha sido realizada por el intermediario.

El valor predeterminado es XMSC_WMQ_MSEL_CLIENT.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir. La selección de mensajes del intermediario no está soportada si la propiedad XMSC_WMQ_BROKER_VERSION está establecida en XMSC_WMQ_BROKER_V1.

XMSC_WMQ_MSG_BATCH_SIZE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El número máximo de mensajes que se va a recuperar de una cola en un lote cuando se utiliza la entrega de mensajes asíncrona.

Cuando una aplicación está utilizando la entrega de mensajes asíncrona, bajo determinadas condiciones, el cliente XMS recupera un lote de mensajes de una cola antes de enviar cada mensaje individualmente a la aplicación. Esta propiedad especifica el número máximo de mensajes que puede haber en un lote.

El valor de la propiedad es un entero positivo y el valor predeterminado es 10. Considere definir la propiedad en un valor diferente solo si tiene un problema de rendimiento específico que debe abordar.

Si una aplicación está conectada a un gestor de colas sobre una red, aumentar el valor de esta propiedad puede reducir las sobrecargas de la red y los tiempos de respuesta, pero puede aumentar la cantidad de memoria necesaria para almacenar mensajes en el sistema cliente. En cambio, si se reduce el valor de esta propiedad se podría aumentar las sobrecargas de la red y los tiempos de respuesta, pero se podría reducir la cantidad de memoria necesaria para almacenar los mensajes.

XMSC_WMQ_POLLING_INTERVAL

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

Si cada escucha de mensajes dentro de una sesión no tiene ningún mensaje adecuado en su cola, este valor es el intervalo máximo, en milisegundos, que transcurre antes de que cada escucha de mensajes intente de nuevo obtener un mensaje de su cola.

Si con frecuencia sucede esto de que no haya disponible ningún mensaje adecuado para ninguno de los escuchas de mensajes de una sesión, considere aumentar el valor de esta propiedad.

El valor de la propiedad es un entero positivo. El valor predeterminado es 5000.

XMSC_WMQ_PORT

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El número del puerto en el cual un gestor de colas escucha solicitudes entrantes.

Esta propiedad solo se utiliza cuando una aplicación se conecta a un gestor de colas en modalidad cliente. La propiedad se utiliza con la propiedad XMSC_WMQ_HOST_NAME para identificar el gestor de colas.

El valor predeterminado de la propiedad es XMSC_WMQ_DEFAULT_CLIENT_PORT, o 1414.

XMSC_WMQ_PROVIDER_VERSION

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

La versión, release, nivel de modificación y fixpack del gestor de colas al que tiene intención conectarse la aplicación. Los valores válidos para esta propiedad son:

- No especificado

O una serie en uno de los formatos siguientes

- V.R.M.F
- V.R.M
- V.R
- V

donde V, R, M y F son valores enteros mayores que o igual a cero.

Un valor de 7 o superior indica que esta versión está pensada como un ConnectionFactory de IBM WebSphere MQ Versión 7.0 para conexiones a un gestor de colas IBM WebSphere MQ Versión 7.0. Un valor inferior a 7 (por ejemplo "6.0.2.0"), indica que está pensado para utilizarlo con gestores de colas anteriores a la Versión 7.0. El valor predeterminado, sin especificar, permite conexiones a cualquier nivel de gestor de colas, determinando las propiedades aplicables y las funciones disponibles basándose en las prestaciones del gestor de colas.

De forma predeterminada, esta propiedad está establecida en "sin especificar".

Nota:

- No se produce ninguna compartición de socket si XMSC_WMQ_PROVIDER_VERSION se establece en 6.2.
- La conexión falla si XMSC_WMQ_PROVIDER_VERSION está establecida en 7 y en el servidor SHARECNV para el canal está establecida en 0.
- Las características específicas de IBM WebSphere MQ Versión 7.0 están inhabilitadas si XMSC_WMQ_PROVIDER_VERSION está establecida en UNSPECIFIED y SHARECNV en 0.

La versión del cliente IBM WebSphere MQ también desempeña un papel principal con respecto a si una aplicación cliente XMS puede utilizar características específicas de IBM WebSphere MQ Versión 7.0. La tabla siguiente describe el comportamiento.

Nota: Una propiedad de sistema XMSC_WMQ_OVERRIDEPROVIDERVERSION sustituye la propiedad XMSC_WMQ_PROVIDER_VERSION. Esta propiedad se puede utilizar si no puede cambiar el valor de la fábrica de conexiones.

Tabla 37. Cliente XMS - Capacidad para utilizar características específicas de IBM WebSphere MQ Versión 7.0.

#	XMSC_WMQ_PROVIDER_VERSION	Versión del cliente de IBM WebSphere MQ	Características de IBM WebSphere MQ Versión 7.0
1	no especificada	7	ACTIVADA
2	no especificada	6	DESACTIVADA

Tabla 37. Cliente XMS - Capacidad para utilizar características específicas de IBM WebSphere MQ Versión 7.0. (continuación)

#	XMSC_WMQ_PROVIDER_VERSION	Versión del cliente de IBM WebSphere MQ	Características de IBM WebSphere MQ Versión 7.0
3	7	7	ACTIVADA
4	7	6	Excepción
5	6	6	DESACTIVADA
6	6	7	DESACTIVADA

XMSC_WMQ_PUB_ACK_INTERVAL

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

Número de mensajes publicados por una aplicación de publicación antes de que el cliente XMS solicite un acuse de recibo al intermediario.

Si reduce el valor de esta propiedad, el cliente solicita acuses de recibo con más frecuencia y, por lo tanto, el rendimiento del publicador disminuye. Si se aumenta el valor, el cliente tarda más tiempo en emitir una excepción si el intermediario no se ejecuta correctamente.

El valor de la propiedad es un entero positivo. El valor predeterminado es 25.

XMSC_WMQ_QMGR_CCSID

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El identificador (CCSID) del juego de caracteres codificado, o página de códigos, en el que se intercambian los campos de datos de caracteres definidos en la interfaz de cola de mensajes (MQI) entre el cliente de XMS y el cliente de IBM WebSphere MQ . Esta propiedad no se aplica a las series de datos de caracteres en los cuerpos de mensajes.

Cuando la aplicación Un XMS se conecta a un gestor de colas en la modalidad de cliente, el cliente de XMS se enlaza al cliente de IBM WebSphere MQ. La información intercambiada entre los dos clientes contiene campos de datos de caracteres que están definidos en la MQI. Bajo circunstancias normales, el cliente de IBM WebSphere MQ da por supuesto que estos campos están en la página de códigos del sistema en el cual se están ejecutando los clientes. Si el cliente de XMS proporciona y espera recibir estos campos en una página de códigos distinta, debe establecer esta propiedad para informar de ello al cliente de IBM WebSphere MQ.

Cuando el cliente de IBM WebSphere MQ envía estos campos de datos de caracteres al gestor de colas, los datos incluidos se deben convertir, si es necesario, a la página de códigos utilizada por el gestor de colas. De forma similar, cuando el cliente de IBM WebSphere MQ recibe estos campos del gestor de colas, los datos aquí incluidos se deben convertir, si es necesario, a la página de códigos en la cual el cliente de XMS espera recibir los datos. El cliente de IBM WebSphere MQ utiliza esta propiedad para realizar estas conversiones de datos.

De forma predeterminada, la propiedad no está establecida.

Establecer esta propiedad es equivalente a establecer la variable de entorno MQCCSID para un cliente de IBM WebSphere MQ que da soporte a aplicaciones cliente de IBM WebSphere MQ nativas. Si desea más información sobre esta variable de entorno, consulte *IBM WebSphere MQ Clientes*.

XMSC_WMQ_QUEUE_MANAGER

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre del gestor de colas al que conectarse.

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_RECEIVE_CCSID

La propiedad de destino que define el CCSID de destino para la conversión de mensajes del gestor de colas. El valor se ignora, a menos que XMSC_WMQ_RECEIVE_CONVERSION se establezca en WMQ_RECEIVE_CONVERSION_QMGR.

Tipo de datos:
Entero

Valor:
Cualquier entero positivo.

El valor predeterminado es 1208.

XMSC_WMQ_RECEIVE_CONVERSION

La propiedad de destino que determina si el gestor de colas va a realizar la conversión de datos.

Tipo de datos:
Entero

Valores:
XMSC_WMQ_RECEIVE_CONVERSION_CLIENT_MSG (VALOR PREDETERMINADO): Realizar la conversión de datos solo en el cliente XMS. La conversión siempre se realiza utilizando la página de códigos 1208.

XMSC_WMQ_RECEIVE_CONVERSION_QMGR: Realizar conversión de datos en el gestor de colas antes de enviar un mensaje al cliente XMS.

XMSC_WMQ_RECEIVE_EXIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Identifica una salida de recepción de canal que se va a ejecutar.

El valor de la propiedad es una serie que identifica una salida de recepción de canal y tiene el formato siguiente:

libraryName(nombrepuntoentrada)

Donde:

- **libraryName** es la vía de acceso completa del archivo `.dll` de salida gestionada
- `nombrepuntoentrada` es el nombre de clase calificado por el espacio de nombres

Por ejemplo, `C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)`

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado. Asimismo, solo están soportadas las salidas gestionadas.

XMSC_WMQ_RECEIVE_EXIT_INIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Los datos de usuario que se pasan a una salida de recepción de canal cuando se llama.

El valor de la propiedad es una serie. De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado y la propiedad "XMSC_WMQ_RECEIVE_EXIT" en la página 236 está establecida.

XMSC_WMQ_RESOLVED_QUEUE_MANAGER

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Esta propiedad se utiliza para obtener el nombre del gestor de colas que está conectado.

Cuando se utiliza con una CCDT (tabla de definición de canal de cliente), este nombre podría ser diferente del nombre del gestor de colas especificado en la fábrica de conexiones.

XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Esta propiedad se llena con el ID del gestor de colas después de la conexión.

XMSC_WMQ_SECURITY_EXIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Identifica una salida de seguridad de canal.

El valor de la propiedad es una serie que identifica una salida de seguridad de canal y tiene el formato siguiente:

libraryName(nombrepuntoentrada)

Donde:

- **libraryName** es la vía de acceso completa del archivo .dll de salida gestionada.
- nombrepuntoentrada es el nombre de clase calificado por el espacio de nombres

Por ejemplo, C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

La longitud máxima de la serie es 128 caracteres.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado. Asimismo, solo están soportadas las salidas gestionadas.

XMSC_WMQ_SECURITY_EXIT_INIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Los datos de usuario que se pasan a una salida de seguridad de canal cuando se llama.

La longitud máxima de la serie de datos de usuario es 32 caracteres.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado y la propiedad "XMSC_WMQ_SECURITY_EXIT" en la página 237 está establecida.

XMSC_WMQ_SEND_EXIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Identifica una salida de emisión de canal.

El valor de la propiedad es una serie. Una salida de emisión de canal tiene el formato siguiente:

libraryName(nombrepuntoentrada)

Donde:

- **libraryName** es la vía de acceso completa del archivo .dll de salida gestionada.
- nombrepuntoentrada es el nombre de clase calificado por el espacio de nombres

Por ejemplo, `C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)`

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado. Asimismo, solo están soportadas las salidas gestionadas.

XMSC_WMQ_SEND_EXIT_INIT

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Los datos de usuario que se pasan a las salidas de emisión de canal cuando se invocan.

El valor de la propiedad es una serie de uno o más elementos de datos de usuario separados por comas. De forma predeterminada, la propiedad no está establecida.

Las reglas para especificar datos de usuario que se pasan a una secuencia de salidas de emisión de canal son las mismas que las reglas para especificar datos de usuario que se pasan a una secuencia de salidas de recepción de canal. Por lo tanto, para las reglas consulte "XMSC_WMQ_RECEIVE_EXIT_INIT" en la página 237.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente gestionado y la propiedad "XMSC_WMQ_SEND_EXIT" en la página 238 está establecida.

XMSC_WMQ_SEND_CHECK_COUNT

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El número de llamadas de envío que se van a permitir entre las comprobaciones de errores de colocación asíncrona dentro de una sesión XMS única sin transacción.

De forma predeterminada, esta propiedad está establecida en 0.

XMSC_WMQ_SHARE_CONV_ALLOWED

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

Indica si una conexión de cliente puede compartir su socket con otras conexiones XMS de nivel superior del mismo proceso con el mismo gestor de colas, si las definiciones de canal coinciden. Esta propiedad se proporciona para permitir un aislamiento completo de conexiones en sockets separados si es necesario para el desarrollo de aplicaciones, el mantenimiento o por motivos operativos. Definir esta propiedad simplemente indica a XMS que se comparta el socket subyacente. No indica cuántas conexiones comparten un solo socket. El número de conexiones que comparten un socket se determina mediante el valor SHARECNV que se negocia entre el cliente de IBM WebSphere MQ y el servidor IBM WebSphere MQ.

Una aplicación puede establecer las constantes con nombre siguiente para establecer la propiedad:

- XMSC_WMQ_SHARE_CONV_ALLOWED_FALSE - Las conexiones no comparten un socket.
- XMSC_WMQ_SHARE_CONV_ALLOWED_TRUE - Las conexiones comparten un socket.

De forma predeterminada, la propiedad está establecida en XMSC_WMQ_SHARE_CONV_ALLOWED_ENABLED.

Esta propiedad solo es relevante cuando una aplicación se conecta a un gestor de colas en modalidad de cliente.

XMSC_WMQ_SSL_CERT_STORES

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Las ubicaciones de los servidores que contienen las listas de revocaciones de certificados (CRL) que se van a utilizar en una conexión SSL a un gestor de colas.

El valor de la propiedad es una lista de uno o más URL separados por comas. Cada URL tiene el formato siguiente:

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

Este formato es compatible con, pero con ampliaciones, el formato MQJMS básico.

Es válido tener un `serveraddress`vacío. En este caso, XMS presupone que el valor es la serie "localhost".

Una lista de ejemplos es:

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

Sólo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_SSL_CIPHER_SPEC

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

V 7.5.0.2 El nombre de la CipherSpec que se va a utilizar en una conexión segura a un gestor de colas.

Las especificaciones de cifrado que puede utilizar con el soporte de SSL y TLS de IBM WebSphere MQ aparecen listadas en la tabla siguiente. Cuando solicite un certificado personal, especifique un tamaño de clave para el par de claves pública y privada. El tamaño de clave que se utiliza durante el reconocimiento SSL es el tamaño almacenado en el certificado, a menos que esté determinado por la CipherSpec, tal como está indicado en la tabla. De forma predeterminada, esta propiedad no está establecida.

Nombre de CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de cifrado	Bits de cifrado	FIPS[1]	Suite B de 128 bits	Suite B de 192 bits
NULL_MD5	SSL 3.0 [8]	MD5	Ninguna	0	No	No	No
NULL_SHA	SSL 3.0 [8]	SHA-1	Ninguna	0	No	No	No
RC4_MD5_EXPORT [2]	SSL 3.0 [8]	MD5	RC4	40	No	No	No
RC4_MD5_US	SSL 3.0 [8]	MD5	RC4	128	No	No	No
RC4_SHA_US	SSL 3.0 [8]	SHA-1	RC4	128	No	No	No
RC2_MD5_EXPORT[2]	SSL 3.0 [8]	MD5	RC2	40	No	No	No
DES_SHA_EXPORT [2]	SSL 3.0 [8]	SHA-1	DES	56	No	No	No
RC4_56_SHA_EXPORT1024[3]	SSL 3.0 [8]	SHA-1	RC4	56	No	No	No
DES_SHA_EXPORT1024 [3]	SSL 3.0 [8]	SHA-1	DES	56	No	No	No
TRIPLE_DES_SHA_US	SSL 3.0 [8]	SHA-1	3DES	168	No	No	No
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	AES	128	Sí	No	No
TLS_RSA_WITH_AES_256_CBC_SHA[4]	TLS 1.0	SHA-1	AES	256	Sí	No	No
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	DES	56	No[5]	No	No
TLS_RSA_WITH_3DES_EDE_CBC_SHA[8]	TLS 1.0	SHA-1	3DES	168	Sí	No	No
FIPS_WITH_DES_CBC_SHA	SSL 3.0	SHA-1	DES	56	No[6]	No	No

Nombre de CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de cifrado	Bits de cifrado	FIPS[1]	Suite B de 128 bits	Suite B de 192 bits
FIPS_WITH_3DES_EDE_CBC_SHA	SSL 3.0	SHA-1	3DES	168	No[7]	No	No
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Sí	No	No
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Sí	No	No
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Sí	No	No
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	AES	256	Sí	No	No
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	No	No	No
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Sí	No	No
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	No	No	No
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Sí	No	No
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Sí	No	No
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Sí	No	No
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Sí	No	No
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Sí	No	No
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Sí	Sí	No
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Sí	No	Sí
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Sí	No	No
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Sí	No	No
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	Ninguna	0	No	No	No
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	Ninguna	0	No	No	No
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	Ninguna	0	No	No	No
TLS_RSA_WITH_NULL_NULL	TLS 1.2	Ninguna	Ninguna	0	No	No	No

Nombre de CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de cifrado	Bits de cifrado	FIPS[1]	Suite B de 128 bits	Suite B de 192 bits
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	No	No	No

Notas:

1. Especifica si la CipherSpec es compatible con los estándares federales de procesamiento de información (FIPS) 140-2. Para obtener una explicación de FIPS e información sobre cómo configurar WebSphere MQ para el funcionamiento compatible con FIPS 140-2, consulte *Federal Information Processing Standards (FIPS)* en la documentación del producto en línea de IBM IBM WebSphere MQ .
2. El tamaño máximo de la clave de reconocimiento es de 512 bits. Si cualquiera de los certificados intercambiados durante el reconocimiento SSL tiene un tamaño de clave mayor de 512 bits, se genera una clave temporal de 512 bits para poder utilizarla durante el reconocimiento.
3. El tamaño de clave de reconocimiento es de 1024 bits.
4. Este CipherSpec no se puede utilizar para garantizar una conexión desde WebSphere MQ Explorer a un gestor de colas amenos que se apliquen los archivos de políticas no restringidas apropiados al JRE utilizado por Explorer.
5. Esta CipherSpec obtuvo el certificado FIPS 140-2 antes del 19 mayo de 2007.
6. Esta CipherSpec obtuvo el certificado FIPS 140-2 antes del 19 mayo de 2007. El nombre FIPS_WITH_DES_CBC_SHA es histórico y refleja el hecho de que este CipherSpec era anteriormente (pero ya no lo es) compatible con FIPS. Esta CipherSpec está en desuso.
7. El nombre FIPS_WITH_3DES_EDE_CBC_SHA es histórico y refleja el hecho de que este CipherSpec era anteriormente (pero ya no lo es) compatible con FIPS. Esta CipherSpec está en desuso.
8. Cuando se configura WebSphere MQ para que su funcionamiento sea compatible con FIPS 140-2, se puede utilizar esta CipherSpec para transferir hasta 32 GB de datos antes de que la conexión concluya con el error AMQ9288. Para evitar este error, evite utilizar el triple DES (que está en desuso), o habilite el restablecimiento de clave secreta al utilizar esta CipherSpec en una configuración de FIPS 140-2.

Conceptos relacionados

Seguridad

Integridad de datos de mensajes

Tareas relacionadas

Especificación de CipherSpecs

XMSC_WMQ_SSL_CIPHER_SUITE

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de la CipherSuite que se va a utilizar en una conexión SSL o TLS con un gestor de colas. El protocolo utilizado en la negociación de la conexión segura depende de la CipherSuite especificada.

Esta propiedad tiene los valores canónicos siguientes:

- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
- SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_FIPS_WITH_DES_CBC_SHA
- SSL_RSA_WITH_NULL_MD5

- SSL_RSA_WITH_NULL_SHA
- SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
- SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
- SSL_RSA_EXPORT_WITH_RC4_40_MD5
- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_AES_128_CBC_SHA
- SSL_RSA_WITH_AES_256_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA

Este valor se puede proporcionar como alternativa a XMSC_WMQ_SSL_CIPHER_SPEC.

Si se ha especificado un valor no vacío para XMSC_WMQ_SSL_CIPHER_SPEC, este valor sustituye el valor para XMSC_WMQ_SSL_CIPHER_SUITE. Si XMSC_WMQ_SSL_CIPHER_SPEC no tiene un valor, el valor de XMSC_WMQ_SSL_CIPHER_SUITE se utiliza como la suite de cifrado que se va a proporcionar a GSKit. En este caso, el valor se correlaciona en el valor de CipherSpec equivalente, tal como se describe en "Correlaciones de nombres CipherSuite y CipherSpec para conexiones a un IBM WebSphere MQgestor de colas" en la página 68.

Si tanto XMSC_WMQ_SSL_CIPHER_SPEC como XMSC_WMQ_SSL_CIPHER_SUITE están vacíos, el campo pChDef->SSLCipherSpec se llena con espacios.

Sólo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no soportarán conexiones SSL, pero podrían estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_SSL_CRYPTO_HW

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Detalles de configuración para el hardware de cifrado conectado al sistema cliente.

Esta propiedad tiene los valores canónicos siguientes:

- GSK_ACCELERATOR_RAINBOW_CS_OFF
- GSK_ACCELERATOR_RAINBOW_CS_ON
- GSK_ACCELERATOR_NCIPHER_NF_OFF
- GSK_ACCELERATOR_NCIPHER_NF_ON

Existe un formato especial para el hardware de cifrado PKCS11 (donde DriverPath, TokenLabel y TokenPassword son series especificadas por usuario):

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS no interpreta ni altera el contenido de la serie. Copia el valor proporcionado, hasta un límite de 256 caracteres de byte único, en el campo MQSCO.CryptoHardware.

Solo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_SSL_FIPS_REQUIRED

Tipo de datos:
Boolean

Propiedad de:
ConnectionFactory

El valor de esta propiedad determina si una aplicación puede o no puede utilizar suites de cifrado no compatibles con FIPS. Si esta propiedad se establece en true (verdadero), solo se utilizan algoritmos FIPS para la conexión cliente/servidor.

Esta propiedad puede tener los valores siguientes, que se convierten a los dos valores canónicos para MQSCO.FipsRequired:

Tabla 38. Tabla de valores para la propiedad MQSCO.FlipsRequired

Valor	Descripción	Valor correspondiente de MQSCO.FipsRequired
falso	Se puede utilizar cualquier CipherSpec.	MQSSL_FIPS_NO (el valor predeterminado)
true	Solo se pueden utilizar algoritmos criptográficos con certificado FIPS en la CipherSpec que se aplica a esta conexión de cliente.	MQSSL_FIPS_YES

XMS copia el valor relevante en MQSCO.FipsRequired antes de llamar a MQCONNX.

El parámetro MQSCO.FipsRequired sólo está disponible desde IBM WebSphere MQ versión 6. Si IBM WebSphere MQ versión 5.3, si esta propiedad está establecida, XMS no intenta establecer la conexión con el gestor de colas y lanza una excepción adecuada en su lugar.

Solo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

XMSC_WMQ_SSL_KEY_REPOSITORY

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

La ubicación de un archivo de base de datos de claves en el cual se almacenan claves y certificados.

XMS copia la serie, hasta un límite de 256 caracteres de byte único, en el campo MQSCO.KeyRepository. IBM WebSphere MQ interpreta esta serie como un nombre de archivo, incluida la vía de acceso completa.

Solo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a las conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_SSL_KEY_RESETCOUNT

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El KeyResetCount representa el número total de bytes sin cifrado enviados y recibidos en una conversación SSL antes de renegociar la clave secreta. El número de bytes incluye la información de control que envía el MCA.

XMS copia el valor que proporcione para esta propiedad en MQSCO.KeyResetCount antes de llamar a MQCONNX.

El parámetro MQSCO.KeyRestCount solo está disponible desde IBM WebSphere MQ versión 6. Si IBM WebSphere MQ versión 5.3, si esta propiedad está establecida, XMS no intenta establecer la conexión con el gestor de colas y lanza una excepción adecuada en su lugar.

Solo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a las conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

El valor predeterminado de esta propiedad es cero, lo que significa que las claves secretas nunca se renegocian.

XMSC_WMQ_SSL_PEER_NAME

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de igual que se va a utilizar en una conexión SSL a un gestor de colas.

No hay ninguna lista de valores canónicos para esta propiedad. En su lugar, debe crear esta serie de acuerdo con las reglas para SSLPEER.

Un ejemplo de un nombre de igual es:

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS copia la serie en la página de códigos de byte único y coloca los valores correctos en MQCD.SSLPeerNamePtr y MQCD.SSLPeerNameLength antes de llamar a MQCONNX.

Esta propiedad solo es relevante si la aplicación se conecta a un gestor de colas en modalidad de cliente.

Solo para .NET : las conexiones gestionadas con IBM WebSphere MQ (WMQ_CM_CLIENT) no dan soporte a conexiones SSL, pero pueden estar soportadas utilizando una conexión no gestionada (WMQ_CM_CLIENT_UNMANAGED).

De forma predeterminada, la propiedad no está establecida.

XMSC_WMQ_SYNCPOINT_ALL_GETS

Tipo de datos:
System.Boolean

Propiedad de:
ConnectionFactory

Indica si se deben recuperar todos los mensajes de colas dentro del control de punto de sincronización.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
falso	Cuando las circunstancias son apropiadas, el cliente de XMS puede recuperar mensajes de colas fuera del control de punto de sincronización.
true	El cliente de XMS debe recuperar todos los mensajes de las colas dentro del control de punto de sincronización.

El valor predeterminado es false.

XMSC_WMQ_TARGET_CLIENT

Tipo de datos:
System.Int32

Propiedad de:
Destino

Nombre utilizado en un URI:
targetClient

Indica si los mensajes enviados al destino contienen una cabecera MQRFH2.

Si una aplicación envía un mensaje que contiene una cabecera MQRFH2, la aplicación receptora debe poder manejar la cabecera.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WMQ_TARGET_DEST_JMS	Los mensajes enviados al destino contienen una cabecera MQRFH2. Especifique este valor si la aplicación está enviando los mensajes a otra aplicación XMS, una aplicación WebSphere JMS o una aplicación IBM WebSphere MQ nativa que se ha diseñado para manejar una cabecera MQRFH2.
XMSC_WMQ_TARGET_DEST_MQ	Los mensajes enviados al destino no contienen una cabecera MQRFH2. Especifique este valor si la aplicación está enviando los mensajes a una aplicación IBM WebSphere MQ nativa que no se ha diseñado para manejar una cabecera MQRFH2.

El valor predeterminado es XMSC_WMQ_TARGET_DEST_JMS.

XMSC_WMQ_TEMP_Q_PREFIX

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El prefijo utilizado para formar el nombre de la cola dinámica IBM WebSphere MQ que se crea cuando la aplicación crea la cola temporal Un XMS .

Las reglas para formar el prefijo son las mismas que las reglas para formar el contenido del campo **DynamicQName** en un descriptor de objeto, pero el último carácter no en blanco debe ser un asterisco (*). Si la propiedad no está establecida, el valor utilizado es `CSQ.*` en z/OS y `AMQ.*` en las otras plataformas. De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante en el dominio de punto a punto.

XMSC_WMQ_TEMP_TOPIC_PREFIX

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory, Destination

Al crear temas temporales, XMS genera una serie de tema con el formato "`TEMP/TEMPTOPICPREFIX/unique_id`", o si esta propiedad se deja con el valor predeterminado, sólo "`TEMP/unique_id`". Si se especifica un valor no vacío se permite que se definan colas de modelo específicas para crear las colas gestionadas para suscriptores de temas temporales creados en esta conexión.

Cualquier serie no nula que solo está formada por caracteres válidos para una serie de tema de IBM WebSphere MQ es un valor válido para esta propiedad.

De forma predeterminada, esta propiedad está establecida en "" (serie vacía).

Nota: Esta propiedad solo es relevante en el dominio de publicación/suscripción.

XMSC_WMQ_TEMPORARY_MODEL

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Nombre de la cola modelo de IBM WebSphere MQ a partir de la cual se crea una cola dinámica cuando la aplicación crea una cola temporal de Un XMS .

El valor predeterminado de la propiedad es SYSTEM.DEFAULT.MODEL.QUEUE.

Esta propiedad solo es relevante en el dominio de punto a punto.

XMSC_WMQ_WILDCARD_FORMAT

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory, Destination

Esta propiedad determina qué versión de sintaxis de comodín se va a utilizar.

Cuando se utiliza la publicación/suscripción con IBM WebSphere MQ '*' y '?' se tratan como comodines. Considerando que '#' y '+' se tratan como comodines cuando se utiliza la publicación/suscripción con WebSphere Message Broker. Esta propiedad sustituye la propiedad XMSC_WMQ_BROKER_VERSION.

Los valores válidos para esta propiedad son:

XMSC_WMQ_WILDCARD_TOPIC_ONLY
Reconoce sólo los comodines de nivel de tema, es decir, '#' y '+' se tratan como comodines. Este valor es el mismo que XMSC_WMQ_BROKER_V2.

XMSC_WMQ_WILDCARD_CHAR_ONLY
Reconoce sólo los caracteres comodín, es decir, "*" y '?' se tratan como comodines. Este valor es el mismo que XMSC_WMQ_BROKER_V1.

De forma predeterminada, esta propiedad se establece en XMSC_WMQ_WILDCARD_TOPIC_ONLY.

Nota: Esta propiedad no es relevante cuando se realiza la publicación/suscripción utilizando IBM WebSphere MQ Versión 6.0 y anteriores. En su lugar, debe utilizarse la propiedad XMSC_WMQ_BROKER_VERSION.

XMSC_WPM_BUS_NAME

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory y Destination

Nombre utilizado en un URI:
busName

Para una fábrica de conexiones, el nombre del bus de integración de servicios al que se conecta la aplicación o, para un destino, el nombre del bus de integración de servicios en el cual existe el destino.

Para un destino que es un tema, esta propiedad es el nombre del bus de integración de servicios en el cual existe el espacio de tema asociado. Este espacio de tema se especifica mediante la propiedad XMSC_WPM_TOPIC_SPACE.

Si la propiedad no está establecida para un destino, se da por supuesto que la cola o el espacio de tema asociado existe en el bus de integración de servicios al cual se conecta la aplicación.

De forma predeterminada, la propiedad no está establecida.

XMSC_WPM_CONNECTION_PROTOCOL

Tipo de datos:
System.Int32

Propiedad de:
Conexión

El protocolo de comunicaciones utilizado para la conexión al motor de mensajería. Esta propiedad es de solo lectura.

Los valores posibles de la propiedad son solo siguientes:

Valor	Significado
XMSC_WPM_CP_HTTP	La conexión utiliza HTTP sobre TCP/IP.
XMSC_WPM_CP_TCP	La conexión utiliza TCP/IP.

XMSC_WPM_CONNECTION_PROXIMITY

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El valor de proximidad de conexión para la conexión. Esta propiedad determina cómo de cerca debe estar el motor de mensajería al que se conecta la aplicación en el servidor de programa de arranque.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Valor de proximidad de conexión
XMSC_WPM_CONNECTION_PROXIMITY_BUS	Autobús
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	Clúster
XMSC_WPM_CONNECTION_PROXIMITY_HOST	Host
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	Servidor

El valor predeterminado es XMSC_WPM_CONNECTION_PROXIMITY_BUS.

XMSC_WPM_DUR_SUB_HOME

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Nombre utilizado en un URI:
durableSubscriptionHome

El nombre del motor de mensajería donde se gestionan todas las suscripciones duraderas para una conexión o destino. Los mensajes que se van a entregar a los suscriptores duraderos se almacenan en el punto de publicación del mismo motor de mensajería.

Se debe especificar un inicio de suscripción duradera para una conexión antes de que una aplicación pueda crear una suscriptor duradero que utilice la conexión. Cualquier valor especificado para un destino altera temporalmente el valor especificado para la conexión.

De forma predeterminada, la propiedad no está establecida.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WPM_HOST_NAME

Tipo de datos:
Cadena

Propiedad de:
Conexión

El nombre de host o la dirección IP del sistema que contiene el motor de mensajería al que se conecta la aplicación. Esta propiedad es de solo lectura.

XMSC_WPM_LOCAL_ADDRESS

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Para una conexión a un bus de integración de servicios, esta propiedad especifica la interfaz de red local que se va a utilizar, o el puerto local o el rango de puertos locales que se va a utilizar, o ambos.

El valor de la propiedad es una serie con el formato siguiente:

[*nombre_host*][(*puerto_bajo*)[,*puerto_alto*])]

Los significados de las variables son los siguientes:

nombre_host
El nombre de host o la dirección IP de la interfaz de red local que se va a utilizar para la conexión.

Proporcionar esta información solo es necesario si el sistema en el cual se está ejecutando la aplicación tiene dos o más interfaces de red y tendrá que poder especificar qué interfaz se debe utilizar para la conexión. Si el sistema tiene solo una interfaz de red, solo se puede utilizar dicha interfaz. Si el sistema tiene dos o más interfaces de red y no especifica qué interfaz se debe utilizar, la interfaz se selecciona de forma aleatoria.

puerto_bajo
El número del puerto local que se va a utilizar para la conexión.

Si *puerto_alto* también se especifica, *puerto_bajo* se interpreta como el número de puerto inferior en un rango de números de puerto.

puerto_alto
El número de puerto superior en un rango de números de puerto. Se debe utilizar uno de los puertos del rango especificado para la conexión.

Aquí aparecen algunos ejemplos de valores válidos de la propiedad:

JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

De forma predeterminada, la propiedad no está establecida.

XMSC_WPM_ME_NAME

Tipo de datos:
Cadena

Propiedad de:
Conexión

El nombre del motor de mensajería al que se conecta la aplicación. Esta propiedad es de solo lectura.

XMSC_WPM_NON_PERSISTENT_MAP

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El nivel de fiabilidad de los mensajes no persistente que se envían utilizando la conexión.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Nivel de fiabilidad
XMSC_WPM_MAPPING_AS_DESTINATION	Se determina mediante el nivel de fiabilidad predeterminado especificado para el espacio de cola o tema en el bus de integración de servicios.
XMSC_WPM_MAPPING_BEST_EFFORT_NON_PERSISTENT	Mejor esfuerzo no persistente
XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT	Express no persistente
XMSC_WPM_MAPPING_RELIABLE_NON_PERSISTENT	Fiable no persistente
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Fiable persistente
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Seguro persistente

El valor predeterminado es XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT.

XMSC_WPM_PERSISTENT_MAP

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El nivel de fiabilidad de los mensajes persistentes que se envían mediante la conexión.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Nivel de fiabilidad
XMSC_WPM_MAPPING_AS_DESTINATION	Se determina mediante el nivel de fiabilidad predeterminado especificado para el espacio de cola o tema en el bus de integración de servicios.
XMSC_WPM_MAPPING_BEST_EFFORT_NON_PERSISTENT	Mejor esfuerzo no persistente

Valor válido	Nivel de fiabilidad
XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT	Express no persistente
XMSC_WPM_MAPPING_RELIABLE_NON_PERSISTENT	Fiable no persistente
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Fiable persistente
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Seguro persistente

El valor predeterminado es XMSC_WPM_MAPPING_RELIABLE_PERSISTENT.

XMSC_WPM_PORT

Tipo de datos:
System.Int32

Propiedad de:
Conexión

El número del puerto en el que estaba a la escucha el motor de mensajería al que está conectada la aplicación. Esta propiedad es de solo lectura.

XMSC_WPM_PROVIDER_ENDPOINTS

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

Una secuencia de una o más direcciones de punto final de servidores de programa de arranque. Las direcciones de punto final están separadas con comas.

Un servidor de programa de arranque es un servidor de aplicaciones que es responsable de seleccionar el motor de mensajería al que se conecta la aplicación. La dirección de punto final de un servidor de programa de arranque tiene el formato siguiente:

nombre_host:*número_puerto*:*nombre_cadena*

Los significados de los componentes de una dirección de punto final son los siguientes:

nombre_host
El nombre de host o la dirección IP del sistema en el que reside el servidor de programa de arranque. Si no se especifica ningún nombre de host ni ninguna dirección IP, el valor predeterminado es `localhost`.

número_puerto
El número de puerto en el cual escucha el servidor de programa de arranque solicitudes entrantes. Si no se especifica ningún número de puerto, el valor predeterminado es 7276.

nombre_cadena
El nombre de una cadena de transporte del programa de arranque utilizada por el servidor de programa de arranque. Los valores válidos son los siguientes:

Valor válido	Nombre de la cadena de transporte del programa de arranque
XMSC_WPM_BOOTSTRAP_HTTP	BootstrapTunneledMessaging
XMSC_WPM_BOOTSTRAP_HTTPS	BootstrapTunneledSecureMessaging
XMSC_WPM_BOOTSTRAP_SSL	BootstrapSecureMessaging

Valor válido	Nombre de la cadena de transporte del programa de arranque
XMSC_WPM_BOOTSTRAP_TCP	BootstrapBasicMessaging

Si no se especifica ningún nombre, el valor predeterminado esXMSC_WPM_BOOTSTRAP_TCP.

Si no se especifica ninguna dirección de punto final, el valor predeterminado es localhost:7276:BootstrapBasicMessaging.

XMSC_WPM_TARGET_GROUP

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de un grupo de destino de motores de mensajería. La naturaleza del grupo de destino se determina mediante la propiedad XMSC_WPM_TARGET_TYPE.

Establezca esta propiedad si desea restringir la búsqueda de un motor de mensajería a un subgrupo de los motores de mensajería en el bus de integración de servicios. Si desea que la aplicación pueda conectarse a cualquier motor de mensajería del bus de integración de servicios, no establezca esta propiedad.

De forma predeterminada, la propiedad no está establecida.

XMSC_WPM_TARGET_SIGNIFICANCE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

La importancia del grupo de destinos de motores de mensajería.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WPM_TARGET_SIGNIFICANCE_ PREFERRED	Se selecciona un motor de mensajería en el grupo de destinos, si hay uno disponible. De lo contrario, se selecciona un motor de mensajería fuera del grupo de destinos, siempre que esté en el mismo bus de integración de servicios.
XMSC_WPM_TARGET_SIGNIFICANCE_ Obligatorio	El motor de mensajería seleccionado debe estar en el grupo de destinos. Si un motor de mensajería del grupo de destino no está disponible, el proceso de conexión falla.

El valor predeterminado de la propiedad es XMSC_WPM_TARGET_SIGNIFICANCE_PREFERRED.

XMSC_WPM_TARGET_TRANSPORT_CHAIN

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El nombre de la cadena de transporte de entrada que debe utilizar la aplicación para conectarse a un motor de mensajería.

El valor de la propiedad puede ser el nombre de cualquier cadena de transporte de entrada que está disponible en el servidor de aplicaciones que aloja el motor de mensajería. La constante con nombre siguiente se proporciona para una de las cadenas de transporte de entrada predefinidas:

Constante con nombre	Nombre de la cadena de transporte
XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC	InboundBasicMessaging

El valor predeterminado de la propiedad es XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC.

XMSC_WPM_TARGET_TYPE

Tipo de datos:
System.Int32

Propiedad de:
ConnectionFactory

El tipo del grupo de destinos de motores de mensajería. Esta propiedad determina la naturaleza del grupo de destinos identificados por la propiedad XMSC_WPM_TARGET_GROUP.

Los valores válidos de la propiedad son los siguientes:

Valor válido	Significado
XMSC_WPM_TARGET_TYPE_BUSMEMBER	El nombre del grupo de destinos es el nombre de un miembro de bus. El grupo de destinos está formado por todos los motores de mensajería del miembro de bus.
XMSC_WPM_TARGET_TYPE_CUSTOM	El nombre del grupo de destinos es el nombre de un grupo definido por usuario de motores de mensajería. El grupo de destinos es para todos los motores de mensajería que están registrados con el grupo definido por usuario.
XMSC_WPM_TARGET_TYPE_ME	El nombre del grupo de destinos es el nombre de un motor de mensajería. El grupo de destinos es el motor de mensajería especificado.

De forma predeterminada, la propiedad no está establecida.

XMSC_WPM_TEMP_Q_PREFIX

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El prefijo utilizado para formar el nombre de la cola temporal que se crea en el bus de integración de servicios cuando la aplicación crea la cola temporal de Un XMS. El prefijo puede contener hasta 12 caracteres.

El nombre de una cola temporal empieza con los caracteres "_Q" seguidos por el prefijo. El resto del nombre está formado por caracteres generados por el sistema.

De forma predeterminada, la propiedad no está establecida, lo que significa que el nombre de una cola temporal no tiene un prefijo.

Esta propiedad solo es relevante en el dominio de punto a punto.

XMSC_WPM_TEMP_TOPIC_PREFIX

Tipo de datos:
Cadena

Propiedad de:
ConnectionFactory

El prefijo utilizado para formar el nombre de un tema temporal que crea la aplicación. El prefijo puede contener hasta 12 caracteres.

El nombre de un tema temporal empieza con los caracteres "_T" seguidos por el prefijo. El resto del nombre está formado por caracteres generados por el sistema.

De forma predeterminada, la propiedad no está establecida, lo que significa que el nombre de un tema temporal no tiene prefijo.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

XMSC_WPM_TOPIC_SPACE

Tipo de datos:
Cadena

Propiedad de:
Destino

Nombre utilizado en un URI:
topicSpace

El nombre del espacio de tema que contiene el tema. Solo un destino que es un tema puede tener esta propiedad.

De forma predeterminada, la propiedad no está establecida, lo que significa que se presupone el espacio de tema predeterminado.

Esta propiedad solo es relevante en el dominio de Publicar/Suscribir.

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los Estados Unidos.

Es posible que IBM no ofrezca los productos, servicios o las características que se tratan en este documento en otros países. Consulte al representante local de IBM para obtener información sobre los productos y servicios disponibles actualmente en su zona. Las referencias a programas, productos o servicios de IBM no pretenden indicar ni implicar que sólo puedan utilizarse los productos, programas o servicios de IBM. En su lugar podrá utilizarse cualquier producto, programa o servicio equivalente que no infrinja ninguno de los derechos de propiedad intelectual de IBM. No obstante, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio no IBM.

IBM puede tener patentes o solicitudes de patentes pendientes que cubran el tema principal descrito en este documento. El suministro de este documento no le otorga ninguna licencia sobre estas patentes. Puede enviar consultas sobre licencias, por escrito, a:

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

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el Departamento de propiedad intelectual de IBM de su país o envíe las consultas por escrito a:

Licencias de Propiedad Intelectual
Ley de Propiedad intelectual y legal
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokio 103-8510, Japón

El párrafo siguiente no se aplica al Reino Unido ni a ningún otro país donde estas disposiciones contradigan la legislación vigente: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN NINGÚN TIPO DE GARANTÍA, YA SEA EXPLÍCITA O IMPLÍCITA, INCLUYENDO, PERO SIN LIMITARSE A, LAS GARANTÍAS IMPLÍCITAS DE NO INCUMPLIMIENTO, COMERCIABILIDAD O IDONEIDAD PARA UNA FINALIDAD DETERMINADA. Algunas legislaciones no contemplan la exclusión de garantías, ni implícitas ni explícitas, en determinadas transacciones, por lo que puede haber usuarios a los que no les afecte dicha norma.

Esta información puede contener imprecisiones técnicas o errores tipográficos. La información aquí contenida está sometida a cambios periódicos; tales cambios se irán incorporando en nuevas ediciones de la publicación. IBM puede efectuar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento y sin previo aviso.

Cualquier referencia en esta información a sitios web que no son de IBM se realiza por razones prácticas y de ninguna manera sirve como un respaldo de dichos sitios web. Los materiales de dichos sitios web no forman parte de este producto de IBM y la utilización de los mismos será por cuenta y riesgo del usuario.

IBM puede utilizar o distribuir cualquier información que el usuario le proporcione del modo que considere apropiado sin incurrir por ello en ninguna obligación con respeto al usuario.

Los titulares de licencias de este programa que deseen información del mismo con el fin de permitir: (i) el intercambio de información entre los programas creados de forma independiente y otros programas (incluido este) y (ii) el uso mutuo de la información intercambiada, deben ponerse en contacto con:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N

Rochester, MN 55901
U.S.A.

Dicha información puede estar disponible, sujeta a los términos y condiciones apropiados, incluyendo, en algunos casos, el pago de una cantidad.

El programa bajo licencia que se describe en esta información y todo el material bajo licencia disponible para el mismo lo proporciona IBM bajo los términos del Acuerdo de cliente de IBM, el Acuerdo de licencia de programas internacional de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento incluidos en este documento se han obtenido en un entorno controlado. Por consiguiente, los resultados obtenidos en otros entornos operativos pueden variar de manera significativa. Es posible que algunas mediciones se hayan realizado en sistemas en nivel de desarrollo y no existe ninguna garantía de que estas mediciones serán las mismas en sistemas disponibles generalmente. Además, algunas mediciones pueden haberse estimado por extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deben verificar los datos aplicables a su entorno específico.

La información relativa a productos que no son de IBM se obtuvo de los proveedores de esos productos, sus anuncios publicados u otras fuentes de disponibilidad pública. IBM no ha comprobado estos productos y no puede confirmar la precisión de su rendimiento, compatibilidad o alguna reclamación relacionada con productos que no sean de IBM. Las preguntas relacionadas con las posibilidades de los productos que no sean de IBM deben dirigirse a los proveedores de dichos productos.

Todas las declaraciones relacionadas con una futura intención o tendencia de IBM están sujetas a cambios o se pueden retirar sin previo aviso y sólo representan metas y objetivos.

Este documento contiene ejemplos de datos e informes que se utilizan diariamente en la actividad de la empresa. Para ilustrar los ejemplos de la forma más completa posible, éstos incluyen nombres de personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier similitud con los nombres y direcciones utilizados por una empresa real es puramente casual.

LICENCIA DE COPYRIGHT:

Esta información contiene programas de aplicación de ejemplo en lenguaje fuente que ilustran técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir estos programas de ejemplo de cualquier forma sin pagar ninguna cuota a IBM para fines de desarrollo, uso, marketing o distribución de programas de aplicación que se ajusten a la interfaz de programación de aplicaciones para la plataforma operativa para la que se han escrito los programas de ejemplo. Los ejemplos no se han probado minuciosamente bajo todas las condiciones. IBM, por tanto, no puede garantizar la fiabilidad, servicio o funciones de estos programas.

Puede que si visualiza esta información en copia software, las fotografías e ilustraciones a color no aparezcan.

Información acerca de las interfaces de programación

La información de interfaz de programación, si se proporciona, está pensada para ayudarle a crear software de aplicación para su uso con este programa.

Este manual contiene información sobre las interfaces de programación previstas que permiten al cliente escribir programas para obtener los servicios de IBM WebSphere MQ.

Sin embargo, esta información puede contener también información de diagnóstico, modificación y ajustes. La información de diagnóstico, modificación y ajustes se proporciona para ayudarle a depurar el software de aplicación.

Importante: No utilice esta información de diagnóstico, modificación y ajuste como interfaz de programación porque está sujeta a cambios.

Marcas registradas

IBM, el logotipo de IBM , ibm.com, son marcas registradas de IBM Corporation, registradas en muchas jurisdicciones de todo el mundo. Hay disponible una lista actual de marcas registradas de IBM en la web en "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Otros nombres de productos y servicios pueden ser marcas registradas de IBM o de otras empresas.

Microsoft y Windows son marcas registradas de Microsoft Corporation en EE.UU. y/o en otros países.

UNIX es una marca registrada de Open Group en Estados Unidos y en otros países.

Linux® es una marca registrada de Linus Torvalds en Estados Unidos y en otros países.

Este producto incluye software desarrollado por Eclipse Project (http://www.eclipse.org/).

Java y todas las marcas registradas y logotipos son marcas registradas de Oracle o sus afiliados.

Número Pieza:

(1P) P/N: