

9.0

*IBM Message Service Client for .NET*

**IBM**

**Nota**

Antes de usar estas informações e o produto suportado por elas, leia as informações em [“Avisos” na página 255](#).

Esta edição se aplica à versão 9 liberação 0 do IBM® MQ e a todas as liberações e modificações subsequentes até que seja indicado de outra forma em novas edições.

Ao enviar informações para a IBM, você concede à IBM um direito não exclusivo de usar ou distribuir as informações da maneira que julgar apropriada, sem incorrer em qualquer obrigação para com você

© **Copyright International Business Machines Corporation 2007, 2023.**

---

# Índice

<b>Message Service Client for .NET.....</b>	<b>5</b>
Introdução ao IBM Message Service Client for .NET.....	5
estilos do sistema de mensagens.....	6
O modelo de objeto XMS.....	7
Atributos e Propriedades de Objetos.....	8
Objetos Administrados.....	9
O modelo de mensagem XMS.....	10
Evitando que os aplicativos usem a versão mais recente do XMS.....	12
Configurando o Ambiente do Servidor de Mensagens.....	13
Configurando o gerenciador de filas e o broker para um aplicativo que se conecta a um gerenciador de filas IBM MQ.....	13
Configurando um broker para um aplicativo que usa uma conexão em tempo real com um broker.....	15
Configurando o barramento de integração de serviços para um aplicativo que se conecta ao WebSphere Application Server.....	16
Usando os aplicativos de amostra XMS.....	17
Os aplicativos de amostra:.....	17
Executando os Aplicativos de Amostra.....	19
Construindo os aplicativos de amostra .NET.....	20
Desenvolvendo aplicativos do XMS.....	20
Gravando aplicativos do XMS.....	20
Gravando aplicativos XMS .NET.....	44
Trabalhando com Objetos Administrados.....	50
Protegendo Comunicações para Aplicativos XMS.....	67
Mensagens do XMS.....	70
Resolução de Problemas.....	84
Configuração de rastreo para aplicativos .NET.....	85
Configuração de FFDC para aplicativos .NET.....	89
Dicas para Resolução de Problemas.....	89
Clientes de serviço de mensagens para referência .NET.....	91
.REDE interfaces.....	91
Propriedades de objetos XMS.....	181
<b>Avisos.....</b>	<b>255</b>
Informações sobre a Interface de Programação.....	256
Marcas comerciais.....	257



## Introdução ao IBM Message Service Client for .NET

---

O IBM Message Service Client for .NET fornece uma interface de programação de aplicativos (API) chamada XMS que possui o mesmo conjunto de interfaces que o Java Message Service (JMS) API. O IBM Message Service Client for .NET contém uma implementação totalmente gerenciada do XMS, que pode ser usada por qualquer idioma compatível com o .NET.

O XMS suporta:

- Sistema de mensagens ponto-a-ponto
- Sistema de mensagens Publicação/Assinatura
- Entrega de Mensagem Síncrona
- Entrega de Mensagem Assíncrona

Um aplicativo XMS pode trocar mensagens com os seguintes tipos de aplicativo:

- Um aplicativo do XMS
- Um aplicativo do IBM MQ classes for JMS
- Um aplicativo IBM MQ nativo
- Um aplicativo JMS que está usando o provedor de sistemas de mensagens padrão IBM MQ

Um aplicativo XMS pode se conectar a, e usar os recursos do, qualquer um dos seguintes servidores de sistema de mensagens:

### **Gerenciador de filas da IBM MQ**

O aplicativo pode se conectar no modo de ligações ou de cliente.

### **Barramento de integração de serviços WebSphere Application Server**

O aplicativo pode usar uma conexão TCP/IP direta, ou pode usar HTTP sobre TCP/IP.

### **IBM Integration Bus**

As mensagens são transportadas entre o aplicativo e o broker usando WebSphere MQ Real-time Transport. As mensagens podem ser entregues para o aplicativo usando o WebSphere MQ Multicast Transport.

Ao conectar-se com um gerenciador de filas do IBM MQ, um aplicativo XMS pode usar o WebSphere MQ Enterprise Transport para comunicar-se com o IBM Integration Bus. Como alternativa, um aplicativo XMS pode publicar e assinar pela conexão com IBM MQ.

### **Conceitos relacionados**

[“estilos do sistema de mensagens” na página 6](#)

[“O modelo de objeto XMS” na página 7](#)

A API XMS é uma interface orientada a objetos. O modelo de objeto XMS é baseado no modelo de objeto JMS 1.1 .

[“O modelo de mensagem XMS” na página 10](#)

O modelo de mensagem XMS é o mesmo que o modelo de mensagem IBM MQ classes for JMS .

## Introdução ao IBM Message Service Client for .NET

---

O IBM Message Service Client for .NET fornece uma interface de programação de aplicativos (API) chamada XMS que possui o mesmo conjunto de interfaces que o Java Message Service (JMS) API. O IBM Message Service Client for .NET contém uma implementação totalmente gerenciada do XMS, que pode ser usada por qualquer idioma compatível com o .NET.

O XMS suporta:

- Sistema de mensagens ponto-a-ponto
- Sistema de mensagens Publicação/Assinatura

- Entrega de Mensagem Síncrona
- Entrega de Mensagem Assíncrona

Um aplicativo XMS pode trocar mensagens com os seguintes tipos de aplicativo:

- Um aplicativo do XMS
- Um aplicativo do IBM MQ classes for JMS
- Um aplicativo IBM MQ nativo
- Um aplicativo JMS que está usando o provedor de sistemas de mensagens padrão IBM MQ

Um aplicativo XMS pode se conectar a, e usar os recursos do, qualquer um dos seguintes servidores de sistema de mensagens:

#### **Gerenciador de filas da IBM MQ**

O aplicativo pode se conectar no modo de ligações ou de cliente.

#### **Barramento de integração de serviços WebSphere Application Server**

O aplicativo pode usar uma conexão TCP/IP direta, ou pode usar HTTP sobre TCP/IP.

#### **IBM Integration Bus**

As mensagens são transportadas entre o aplicativo e o broker usando WebSphere MQ Real-time Transport. As mensagens podem ser entregues para o aplicativo usando o WebSphere MQ Multicast Transport.

Ao conectar-se com um gerenciador de filas do IBM MQ, um aplicativo XMS pode usar o WebSphere MQ Enterprise Transport para comunicar-se com o IBM Integration Bus. Como alternativa, um aplicativo XMS pode publicar e assinar pela conexão com IBM MQ.

#### **Conceitos relacionados**

[“estilos do sistema de mensagens” na página 6](#)

[“O modelo de objeto XMS” na página 7](#)

A API XMS é uma interface orientada a objetos. O modelo de objeto XMS é baseado no modelo de objeto JMS 1.1 .

[“O modelo de mensagem XMS” na página 10](#)

O modelo de mensagem XMS é o mesmo que o modelo de mensagem IBM MQ classes for JMS .

## **estilos do sistema de mensagens**

---

O XMS suporta os estilos ponto-a-ponto e Publicação/Assinatura do sistema de mensagens

Estilos de sistema de mensagens também são chamados de domínios de mensagens.

### **Sistema de mensagens Ponto a ponto**

Uma forma comum do sistema de mensagens ponto-a-ponto usa enfileiramento. No caso mais simples, um aplicativo envia uma mensagem para outro aplicativo identificando, implicitamente ou explicitamente, uma fila de destino. O sistema de mensagens subjacente e o sistema de enfileiramento recebem a mensagem do aplicativo de envio e roteia a mensagem para sua fila de destino. O aplicativo de recebimento pode então recuperar a mensagem a partir da fila.

Se o sistema de mensagens e o sistema de enfileiramento subjacentes contiverem o IBM Integration Bus, o IBM Integration Bus poderá replicar uma mensagem e rotear cópias da mensagem para filas diferentes. Como resultado, mais de um aplicativo pode receber a mensagem. IBM Integration Bus também pode transformar uma mensagem e incluir dados a ele.

Uma característica chave do sistema de mensagens do ponto-a-ponto é que um aplicativo coloca uma mensagem em uma fila local quando ele envia uma mensagem. O sistema de mensagens subjacente e o sistema de enfileiramento determinam para qual fila de destino a mensagem é enviada. O aplicativo de recebimento recupera a mensagem da fila de destino.

## Sistema de mensagens Publicar / assinar

No sistema de mensagens do Publicação/Assinatura , há dois tipos de aplicativo: publicador e assinante

Um *publicador* fornece informações sobre a forma de mensagens de publicação. Quando um publicador publica uma mensagem, ele especifica um tópico, que identifica o assunto das informações dentro da mensagem.

Um *assinante* é um consumidor das informações que são publicadas. Um assinante especifica os tópicos em que está interessado pela criação de assinaturas.

O sistema de publicação / assinatura recebe publicações de publicadores e assinaturas de assinantes. Ele roteia publicações para assinantes. Um assinante recebe publicações sobre apenas os tópicos para os quais ele subscreveu.

Uma característica principal do sistema de mensagens do Publicação/Assinatura é que um publicador identifica um tópico quando publica uma mensagem. Ele não identifica os assinantes. Se uma mensagem for publicada em um tópico para o qual não há assinantes, nenhum aplicativo receberá a mensagem.

Um aplicativo pode ser um publicador e um assinante.

## O modelo de objeto XMS

---

A API XMS é uma interface orientada a objetos. O modelo de objeto XMS é baseado no modelo de objeto JMS 1.1 .

A lista a seguir resume as principais classes XMS ou tipos de objeto:

### ConnectionFactory

Um objeto `ConnectionFactory` é encapsulado um conjunto de parâmetros para uma conexão. Um aplicativo usa um `ConnectionFactory` para criar uma conexão. Um aplicativo pode fornecer os parâmetros no tempo de execução e criar um objeto `ConnectionFactory` . Como alternativa, os parâmetros de conexão podem ser armazenados em um repositório de objetos administrados. Um aplicativo pode recuperar um objeto a partir do repositório e criar um objeto `ConnectionFactory` a partir dele.

### Connection

Um objeto `Connection` encapsula uma conexão ativa de um aplicativo para um servidor de sistema de mensagens. Um aplicativo usa uma conexão para criar sessões.

### Destination

Um aplicativo envia mensagens ou recebe mensagens usando um objeto `Destination`. No domínio Publicação/Assinatura , um objeto `Destination` contém um tópico e, no domínio ponto-a-ponto , um objeto `Destination` contém uma fila. Um aplicativo pode fornecer os parâmetros para criar um objeto `Destination` no tempo de execução. Como alternativa, ele pode criar um objeto `Destination` a partir de uma definição de objeto que é armazenada no repositório de objetos administrados.

### Session

Um objeto `Session` é um único contexto encadeado para enviar e receber mensagens. Um aplicativo usa um objeto `Session` para criar objetos `Message`, `MessageProducer` e `MessageConsumer` .

### Message

Um objeto `Message` contém o objeto `Message` que um aplicativo envia usando um objeto `MessageProducer` ou recebe usando um objeto `MessageConsumer`.

### MessageProducer

Um objeto `MessageProducer` é usado por um aplicativo para enviar mensagens para um destino.

### MessageConsumer

Um objeto `MessageConsumer` é usado por um aplicativo para receber mensagens enviadas para um destino.

Figura 1 na página 8 mostra esses objetos e seus relacionamentos. Este diagrama mostra os principais tipos de objeto XMS : `ConnectionFactory`, `Conexão` e `Sessão`, `MessageProducer`, `MessageConsumer` e `Mensagem` e `Destino`. Um aplicativo usa um `connection factory` para criar uma conexão e usa uma

conexão para criar sessões. O aplicativo pode então usar uma sessão para criar mensagens, produtores de mensagens e consumidores de mensagens. O aplicativo usa um produtor de mensagem para enviar mensagens para um destino e usa um consumidor de mensagens para receber mensagens enviadas para um destino.

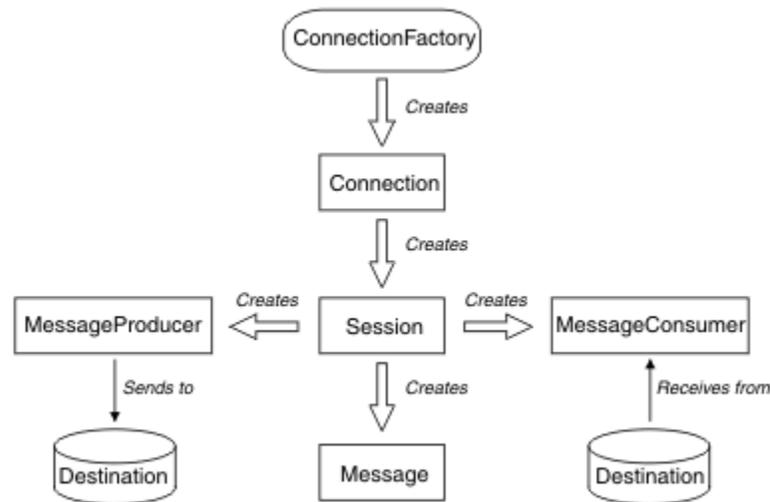


Figura 1. Os objetos do XMS e seus relacionamentos

Em .NET, as classes XMS são definidas como um conjunto de interfaces .NET . Quando você estiver codificando aplicativos do XMS .NET , precisará apenas das interfaces declaradas

O modelo de objeto XMS é baseado nas interfaces independentes de domínio que são descritas em Java Message Service Especificação, Versão 1.1. Classes específicas de domínio, como Topic, TopicPublisher e TopicSubscriber, não são fornecidas.

## Atributos e Propriedades de Objetos

Um objeto XMS pode ter atributos e propriedades, que são características do objeto, que são implementados de diferentes maneiras

### Atributos

Uma característica de objeto que está sempre presente e ocupa o armazenamento, mesmo que o atributo não tenha um valor. Nesse aspecto, um atributo é semelhante a um campo em uma estrutura de dados de comprimento fixo. Um recurso de diferenciação de atributos é que cada atributo possui seus próprios métodos para configurar e obter seu valor.

### Propriedades

Uma propriedade de um objeto está presente e ocupa o armazenamento apenas depois que seu valor for configurado. Uma propriedade não pode ser excluída ou sua memória recuperada após seu valor ser configurado. É possível alterar seu valor. XMS fornece um conjunto de métodos genéricos para a configuração e obtenção de valores de propriedade.

### Conceitos relacionados

#### Tipos primitivos do XMS

O XMS fornece equivalentes dos oito tipos primitivos Java (byte, short, int, long, float, double, char e boolean). Isso permite a troca de mensagens entre o XMS e o JMS sem que os dados se tornem perdidos ou corrompidos.

#### Conversão implícita de um valor de propriedade de um tipo de dados para outro

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

### Referências relacionadas

Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

## Objetos Administrados

Usando objetos administrados, é possível administrar as configurações de conexão usadas pelos aplicativos clientes a serem administradas a partir de um repositório central. Um aplicativo recupera definições de objeto do repositório central e as utiliza para criar objetos `ConnectionFactory` e `Destination`. Usando objetos administrados, é possível desacoplar aplicativos dos recursos usados no tempo de execução.

Por exemplo, os aplicativos XMS podem ser gravados e testados com objetos administrados que fazem referência a um conjunto de conexões e destinos em um ambiente de teste. Quando os aplicativos são implementados, os objetos administrados podem ser alterados para configurar os aplicativos para se referir a conexões e destinos no ambiente de produção.

XMS suporta dois tipos de objeto administrado:

- Um objeto `ConnectionFactory`, que é usado pelos aplicativos para fazer a conexão inicial com o servidor.
- Um objeto `Destination`, que é usado pelos aplicativos para especificar o destino para mensagens que estão sendo enviadas e a origem de mensagens que estão sendo recebidas. Um destino é um tópico ou uma fila no servidor para o qual um aplicativo se conecta.

A ferramenta de administração **JMSAdmin** é fornecida com o IBM MQ. Ela é usada para criar e gerenciar objetos administrados em um armazenador central de objetos administrados.

Os objetos administrados no repositório podem ser usados pelos aplicativos IBM MQ classes for JMS e XMS. Aplicativos XMS podem usar os objetos `ConnectionFactory` e `Destination` para se conectar a um IBM MQ gerenciador de filas. Um administrador pode alterar as definições de objeto mantidas no repositório sem afetar o código do aplicativo.

O diagrama a seguir mostra como um aplicativo XMS geralmente usa objetos administrados. O lado esquerdo do diagrama mostra um repositório contendo as definições de objeto `ConnectionFactory` e `Destination` que são administradas usando um console de administração. O lado direito do diagrama mostra um aplicativo XMS que consulta as definições de objeto no repositório e, em seguida, usa essas definições de objeto ao se conectar a um servidor de sistema de mensagens.

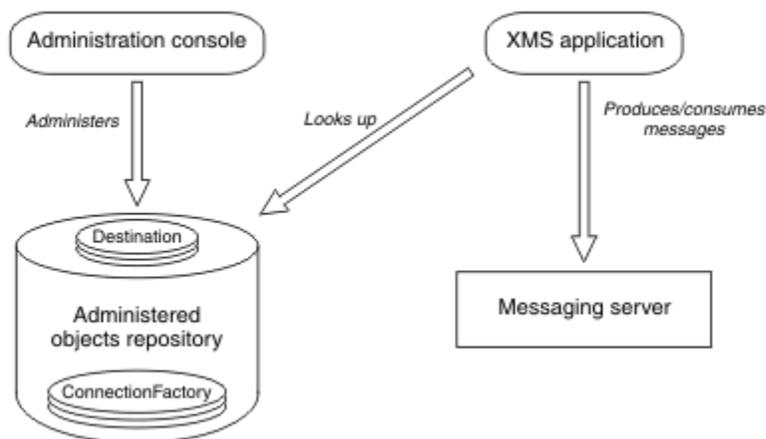


Figura 2. Uso Típico de Objetos Administrados por um Aplicativo XMS

### Conceitos relacionados

[Trabalhando com Objetos Administrados](#)

Os tópicos nesta seção fornecem informações sobre objetos administrados. Os aplicativos XMS podem recuperar as definições de objeto de um repositório de objetos administrados centrais e usá-los para criar connection factories e destinos.

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### **Tarefas relacionadas**

##### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

## O modelo de mensagem XMS

---

O modelo de mensagem XMS é o mesmo que o modelo de mensagem IBM MQ classes for JMS .

Em particular, XMS implementa os campos de cabeçalho da mesma mensagem e as propriedades de mensagem que o IBM MQ classes for JMS implementa:

- Campos de cabeçalho JMS . Esses campos possuem nomes que se iniciam com o prefixo JMS.
- Propriedades definidas pelo JMS. Esses campos possuem propriedades cujos nomes se iniciam com o prefixo JMSX.
- Propriedades definidas pelo IBM. Esses campos possuem propriedades cujos nomes começam com o prefixo JMS\_IBM\_.

Como resultado, os aplicativos XMS podem trocar mensagens com aplicativos IBM MQ classes for JMS . Em cada mensagem, alguns dos campos de cabeçalho e propriedades são configurados pelo aplicativo e outros são configurados por XMS ou IBM MQ classes for JMS. Alguns dos campos configurados por XMS ou IBM MQ classes for JMS são configurados quando a mensagem é enviada e outros quando ela é recebida. Os campos de cabeçalho e propriedades são propagados com uma mensagem por meio de um servidor de sistema de mensagens quando apropriado. Elas são disponibilizadas para qualquer aplicativo que receba a mensagem.

## Instalando Message Service Client for .NET usando o assistente de instalação

---

A instalação usa um instalador MSI InstallShield X/Windows. Duas opções de configuração estão disponíveis, para que seja possível escolher uma instalação completa ou customizada.

### **Sobre esta tarefa**

Para instalar o Message Service Client for .NET em Windows, siga este procedimento

### **Procedimento**

1. Se estiver instalando a partir de um SupportPac , conclua as etapas a seguir, caso contrário, continue diretamente na etapa “2” na página 10.
  - a) No Windows, efetue logon como um administrador
  - b) Execute o instalador do dotNETClientsetup.exe.
2. Aguarde a abertura do assistente de instalação e exiba a mensagem a seguir:

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

Clique em **Avançar**.

O assistente pode solicitar a leitura do contrato de licença.

3. Se você for solicitado a ler o contrato de licença e aceitar os termos do contrato de licença, clique em **Eu aceito os termos no contrato de licença**, em seguida, clique em **Avançar**.

O assistente de instalação solicita que você escolha o tipo de configuração que melhor se adapte às suas necessidades

4. Selecione o tipo de configuração necessário:

- Para instalar todos os recursos do programa e instalá-los no diretório de instalação padrão, clique em **Concluir**.
- Para escolher quais recursos deseja instalar e especificar onde eles estão instalados, clique em **Customizado**.

5. Clique em **Avançar**.

Se você selecionar a opção de instalação completa, o assistente de instalação exibe uma mensagem de que está pronto para iniciar a instalação, conforme descrito na etapa “8” na página 11. Se você selecionar a opção de instalação customizada, o assistente de instalação solicitará que você selecione os recursos que deseja instalar e você deverá concluir a etapa “6” na página 11 e a etapa “7” na página 11 antes de avançar para a etapa “8” na página 11

6. Apenas para uma instalação customizada, clique em um ícone na lista de recursos para especificar quaisquer mudanças em como você deseja que os recursos do Message Service Client for .NET sejam instalados. Se você não quiser instalar o Message Service Client for .NET no diretório sugerido, escolha outro diretório.

Se você optar por instalar o Message Service Client for .NET em um diretório que não existe atualmente, o assistente de instalação criará o diretório para você.

Se desejar desenvolver aplicativos XMS, assegure-se de que o recurso **Ferramentas e Amostras de Desenvolvimento** esteja selecionado. Esse recurso fornece os aplicativos de amostra e as bibliotecas e quaisquer outros arquivos necessários para compilar aplicativos .NET. Se você não selecionar esse recurso, apenas os arquivos necessários para executar aplicativos XMS serão instalados.

7. Se você estiver usando a opção de instalação customizada, clique em **Avançar** após selecionar as opções necessárias conforme descrito na etapa “6” na página 11.

O assistente de instalação exibe uma mensagem de que ele está pronto para iniciar a instalação

8. Clique em **Instalar** para iniciar a instalação.

O assistente de instalação exibe uma barra mostrando o progresso da instalação. Aguarde a conclusão da barra de progresso. Quando a instalação for concluída com êxito, a janela exibirá a seguinte mensagem:

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

9. Clique em **Concluir** para fechar o assistente de instalação.

## Resultados

Você instalou com sucesso o Message Service Client for .NET, que está pronto para ser usado

### Como proceder a seguir

Antes de executar quaisquer aplicativos XMS, incluindo os aplicativos de amostra fornecidos com o XMS, você deve configurar o ambiente do servidor de sistema de mensagens, para obter detalhes, consulte: [“Configurando o Ambiente do Servidor de Mensagens” na página 13.](#)

#### Conceitos relacionados

[Serviço da Web de Consulta de JNDI](#)

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus. Esse serviço

da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Configurando o Ambiente do Servidor de Mensagens

Os tópicos nesta seção descrevem como configurar o ambiente do servidor de sistema de mensagens para permitir que os aplicativos XMS se conectem a um servidor.

#### Usando os aplicativos de amostra XMS

Use os aplicativos de amostra fornecidos com o XMS para verificar sua instalação e configuração do servidor de sistema de mensagens e para ajudá-lo a construir seus aplicativos. As amostras fornecem uma visão geral dos recursos comuns de cada API.

## Pré-requisitos para aplicativos XMS se conectando ao WebSphere MQ

---

Alguns pré-requisitos se aplicam se seu aplicativo XMS se conectar ao WebSphere MQ.

Para aplicativos que se conectam a um gerenciador de filas do WebSphere MQ, deve-se instalar as bibliotecas do cliente WebSphere MQ apropriadas na máquina usada para executar o aplicativo XMS. Essas bibliotecas são pré-instaladas nas máquinas com um gerenciador de fila local

Para clienteXMS para .NET, use as bibliotecas do cliente enviadas com IBM WebSphere MQ 7.0.1.0 ou mais recente. Essas são as classes *WebSphere MQ para .REDE*. Eles ativam conexões do modo cliente para IBM WebSphere MQ 7.0, 6.0e 5.3 gerenciadores de filas e conexões do modo de ligações para um gerenciador de fila local, se ele também for IBM WebSphere MQ 7.0.1.0 ou mais recente

Microsoft .NET Framework 2.0 Pacote Redistribuível deve ser instalado no computador no qual o XMS deve ser instalado. Se esse pacote não estiver disponível, a instalação do XMS falhará. Em seguida, é necessário sair do procedimento de instalação, instalar o Microsoft .NET Framework 2.0 Redistributable Package no computador e executar novamente o procedimento de instalação.

No site de download da Microsoft, é necessário procurar dotnetfx.exe para Microsoft .NET Framework 2.0 Pacote Redistribuível (x86) e NetFx64.exe para Microsoft .NET Framework 2.0 Pacote Redistribuível (x64), o que for aplicável.

#### **Conceitos relacionados**

##### Configurando o Ambiente do Servidor de Mensagens

Os tópicos nesta seção descrevem como configurar o ambiente do servidor de sistema de mensagens para permitir que os aplicativos XMS se conectem a um servidor.

## Evitando que os aplicativos usem a versão mais recente do XMS

---

Por padrão, quando uma versão mais recente do XMS é instalada, os aplicativos que usam a versão anterior alternam automaticamente para a versão mais recente sem precisar recompilar.

#### **Sobre esta tarefa**

O recurso de coexistência de várias versões assegura que a instalação de uma versão mais recente do XMS não sobrescreva a versão XMS anterior. Em vez disso, várias instâncias de montagens de XMS .NET semelhantes coexistem no Global Assembly Cache (GAC), mas possuem números de versão diferentes. Internamente, o GAC usa um arquivo de políticas para rotear as chamadas de aplicativo para a versão mais recente de XMS. Os aplicativos são executados sem uma necessidade de recompilação e podem usar novos recursos disponíveis na versão XMS .NET mais recente.

No entanto, se um aplicativo for necessário para usar a versão mais antiga do XMS, configure o atributo `publisherpolicy` como no no arquivo de configuração do aplicativo

**Nota:** Um arquivo de configuração de aplicativo é um arquivo com um nome que consiste no nome do programa executável para o qual o arquivo está relacionado, com o sufixo `.config`. Por exemplo, o arquivo de configuração de aplicativo para `text.exe` teria o nome `text.exe.config`.

A qualquer momento, no entanto, todos os aplicativos de um sistema usam a mesma versão do XMS .NET.

## Configurando o Ambiente do Servidor de Mensagens

---

Os tópicos nesta seção descrevem como configurar o ambiente do servidor de sistema de mensagens para permitir que os aplicativos XMS se conectem a um servidor.

Para aplicativos que se conectam a um gerenciador de filas IBM MQ , o cliente IBM MQ (ou o gerenciador de filas para o modo de ligações) é necessário.

Atualmente, não há pré-requisitos para aplicativos que usam uma conexão em tempo real com um broker.

Você deve configurar o ambiente do servidor de sistema de mensagens antes de executar quaisquer aplicativos XMS , incluindo os aplicativos de amostra fornecidos com XMS.

Esta seção contém os seguintes tópicos:

- [“Configurando o gerenciador de filas e o broker para um aplicativo que se conecta a um gerenciador de filas IBM MQ” na página 13](#)
- [“Configurando um broker para um aplicativo que usa uma conexão em tempo real com um broker” na página 15](#)
- [“Configurando o barramento de integração de serviços para um aplicativo que se conecta ao WebSphere Application Server” na página 16](#)

### Conceitos relacionados

[Serviço da Web de Consulta de JNDI](#)

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

[Usando os aplicativos de amostra XMS](#)

Use os aplicativos de amostra fornecidos com o XMS para verificar sua instalação e configuração do servidor de sistema de mensagens e para ajudá-lo a construir seus aplicativos. As amostras fornecem uma visão geral dos recursos comuns de cada API.

### Tarefas relacionadas

[Instalando Message Service Client for .NET usando o assistente de instalação](#)

A instalação usa um instalador MSI InstallShield X/Windows. Duas opções de configuração estão disponíveis, para que seja possível escolher uma instalação completa ou customizada.

### Referências relacionadas

[Pré-requisitos para aplicativos XMS se conectando ao WebSphere MQ](#)

Alguns pré-requisitos se aplicam se seu aplicativo XMS se conectar ao WebSphere MQ.

## Configurando o gerenciador de filas e o broker para um aplicativo que se conecta a um gerenciador de filas IBM MQ

Esta seção assume que você está usando IBM WebSphere MQ 7.0.1 ou mais recente. Antes de poder executar um aplicativo que se conecta a um gerenciador de filas IBM MQ , você deve configurar o gerenciador de filas. Para um aplicativo de publicação / assinatura, alguma configuração adicional será necessária se você estiver usando a interface de publicação / assinatura enfileirada.

### Antes de começar

O XMS funciona com o IBM Integration Bus ou o WebSphere Message Broker 6.1 ou mais recente

Antes de iniciar esta tarefa, você executa as seguintes etapas:

- Certifique-se de que seu aplicativo tenha acesso a um gerenciador de filas que está em execução.

- Se seu aplicativo for um aplicativo de publicação / assinatura e usar a interface de publicação / assinatura enfileirada, certifique-se de que o atributo "PSMODE" esteja configurado como "ENABLED" no gerenciador de filas.
- Certifique-se de que seu aplicativo usa um connection factory cujas propriedades são configuradas corretamente para se conectar ao gerenciador de filas. Se o seu aplicativo for um aplicativo de publicação / assinatura, certifique-se de que as propriedades apropriadas do connection factory estejam configuradas para o uso do broker. Para obter mais informações sobre as propriedades de um connection factory, "[Propriedades de ConnectionFactory](#)" na página 182

## Sobre esta tarefa

É possível configurar o gerenciador de filas e o broker para executarem aplicativos XMS da mesma maneira que você configura o gerenciador de filas e a interface de publicação/assinatura enfileirada para executarem aplicativos IBM MQ JMS. As etapas a seguir resumem o que você precisa fazer.

## Procedimento

1. No gerenciador de filas, crie as filas que seu aplicativo precisa.

Para uma visão geral de como você cria filas, consulte [Definindo filas](#).

No caso de ser um aplicativo de publicação/assinatura e usar a interface de publicação/assinatura enfileirada que precisa de acesso às filas do sistema IBM MQ classes for JMS, aguarde até a Etapa 4a antes de criar as filas.

2. Conceda ao ID do usuário associado ao seu aplicativo a autoridade para se conectar ao gerenciador de filas e a autoridade apropriada para acessar as filas.

Para uma visão geral sobre autorização, consulte [Protegendo](#). Se o seu aplicativo se conecta ao gerenciador de filas no modo cliente, consulte também [Clientes e servidores](#).

3. Se o seu aplicativo se conectar ao gerenciador de filas no modo cliente, certifique-se de que um canal de conexão do servidor esteja definido no gerenciador de filas e que um listener seja iniciado.

Você não precisa executar esta etapa para cada aplicativo que se conecta ao gerenciador de filas. Uma definição de canal de conexão do servidor e um listener podem suportar todos os aplicativos que se conectam no modo cliente.

4. Se o seu aplicativo for um aplicativo de publicação / assinatura e usar a interface de publicação / assinatura enfileirada, execute as etapas a seguir.

- a) No gerenciador de filas, crie as filas do sistema IBM MQ classes for JMS executando o script de comandos MQSC que são fornecidos com IBM MQ. Certifique-se de que a ID do usuário associado com o IBM Integration Bus ou WebSphere Message Broker tenha autoridade para acessar as filas.

Para obter informações sobre onde localizar o script e como executá-lo, consulte [Usando IBM MQ classes for Java](#).

Execute esta etapa apenas uma vez para o gerenciador de filas. O mesmo conjunto de filas do sistema IBM MQ classes for JMS pode suportar todos os aplicativos XMS e IBM MQ classes for JMS que se conectam ao gerenciador de filas.

- b) Conceda ao ID do usuário associado ao seu aplicativo a autoridade para acessar as filas do sistema IBM MQ classes for JMS .

Para obter informações sobre quais autoridades o ID do usuário precisa, consulte [Usando IBM MQ classes for JMS](#).

- c) Para um broker de IBM Integration Bus ou WebSphere Message Broker, crie e implemente um fluxo de mensagens para atender a fila na qual os aplicativos enviam mensagens que eles publicam.

O fluxo de mensagens básico inclui um nó de processamento de mensagens MQInput para ler as mensagens publicadas e um nó de processamento de mensagem de publicação para publicar as mensagens.

Para obter informações sobre como criar e implantar um fluxo de mensagens, consulte a documentação do produto IBM Integration Bus ou WebSphere Message Broker disponível a partir do [Página da web da biblioteca de documentação do produto IBM Integration Bus](#).

Você não precisa executar esta etapa se um fluxo de mensagens adequado já estiver implementado no intermediário.

## Resultados

Agora é possível iniciar seu aplicativo.

### Tarefas relacionadas

[Configurando um broker para um aplicativo que usa uma conexão em tempo real com um broker](#)  
Antes de poder executar um aplicativo que usa uma conexão em tempo real com um broker, você deve configurar esse broker.

[Configurando o barramento de integração de serviços para um aplicativo que se conecta ao WebSphere Application Server](#)

Antes de poder executar um aplicativo que se conecta a um barramento de integração de serviços WebSphere Application Server service integration technologies , você deve configurar a integração de serviço da mesma maneira que configurar o barramento de integração de serviços para executar aplicativos JMS que usam o provedor de sistemas de mensagens padrão.

## Configurando um broker para um aplicativo que usa uma conexão em tempo real com um broker

Antes de poder executar um aplicativo que usa uma conexão em tempo real com um broker, você deve configurar esse broker.

### Antes de começar

Antes de iniciar esta tarefa, você executa as seguintes etapas:

- Certifique-se de que seu aplicativo tenha acesso a um broker que está em execução.
- Certifique-se de que seu aplicativo usa um connection factory cujas propriedades são configuradas apropriadamente para uma conexão em tempo real com um broker. Para obter mais informações sobre as propriedades de um connection factory, consulte [“Propriedades de ConnectionFactory” na página 182](#)

### Sobre esta tarefa

Você configura um broker para executar aplicativos XMS da mesma maneira que você configura um broker para executar aplicativos IBM MQ classes for JMS . As etapas a seguir resumem o que você precisa fazer:

### Procedimento

1. Crie e implemente um fluxo de mensagens para ler mensagens a partir da porta TCP/IP na qual um broker está atendendo e publique as mensagens.

Você pode fazer isso de uma das seguintes maneiras:

- Crie um fluxo de mensagens que contenha um nó de processamento de mensagens **Real-timeOptimizedFlow** .
- Crie um fluxo de mensagens que contenha um nó de processamento de mensagens **Real-timeInput** e um nó de processamento de mensagem de publicação.

Você deve configurar o nó **Real-timeOptimizedFlow** ou **Real-timeInput** para atender na porta usada para conexões em tempo real. Em XMS, o número da porta padrão para conexões em tempo real é 1506.

Você não precisa executar esta etapa se um fluxo de mensagens adequado já estiver implementado no intermediário.

2. Se precisar que as mensagens sejam entregues ao seu aplicativo usando o IBM MQ classes for JMS, configure o broker para ativar o multicast. Configure os tópicos que devem ser ativados multicast, especificando uma qualidade de serviço confiável para esses tópicos que requerem multicast confiável.
3. Se o seu aplicativo fornecer um ID de usuário e uma senha quando ele se conectar ao intermediário e você desejar que o intermediário autentique seu aplicativo utilizando essas informações, configure o servidor de nome de usuário e o intermediário para autenticação de senha simples de telnet.

## Resultados

Agora é possível iniciar seu aplicativo.

### Tarefas relacionadas

[Configurando o gerenciador de filas e o broker para um aplicativo que se conecta a um gerenciador de filas IBM MQ](#)

Esta seção assume que você está usando IBM WebSphere MQ 7.0.1 ou mais recente. Antes de poder executar um aplicativo que se conecta a um gerenciador de filas IBM MQ, você deve configurar o gerenciador de filas. Para um aplicativo de publicação / assinatura, alguma configuração adicional será necessária se você estiver usando a interface de publicação / assinatura enfileirada.

[Configurando o barramento de integração de serviços para um aplicativo que se conecta ao WebSphere Application Server](#)

Antes de poder executar um aplicativo que se conecta a um barramento de integração de serviços WebSphere Application Server service integration technologies, você deve configurar a integração de serviço da mesma maneira que configurar o barramento de integração de serviços para executar aplicativos JMS que usam o provedor de sistemas de mensagens padrão.

## Configurando o barramento de integração de serviços para um aplicativo que se conecta ao WebSphere Application Server

Antes de poder executar um aplicativo que se conecta a um barramento de integração de serviços WebSphere Application Server service integration technologies, você deve configurar a integração de serviço da mesma maneira que configurar o barramento de integração de serviços para executar aplicativos JMS que usam o provedor de sistemas de mensagens padrão.

### Antes de começar

Antes de iniciar esta tarefa, você deve executar as seguintes etapas:

- Certifique-se de que um barramento do sistema de mensagens seja criado e que o servidor seja incluído no barramento como um membro do barramento.
- Certifique-se de que seu aplicativo tenha acesso a um barramento de integração de serviços que contém pelo menos um mecanismo de sistema de mensagens que esteja em execução.
- Se a operação HTTP, for necessária, um canal de transporte de entrada do mecanismo do sistema de mensagens HTTP deverá ser definido. Por padrão, os canais para SSL e TCP são definidos durante a instalação do servidor.
- Certifique-se de que seu aplicativo usa um connection factory cujas propriedades são configuradas corretamente para se conectar ao barramento de integração de serviços usando um servidor de autoinicialização. As informações mínimas necessárias são:
  - O terminal do provedor, que descreve o local e o protocolo a serem usados ao negociar uma conexão com o servidor de sistema de mensagens (ou seja, por meio do servidor de auto-inicialização). Em seu formato mais simples, para um servidor instalado com configurações padrão, o terminal de fornecimento pode ser configurado para o nome do host do servidor.
  - O nome do barramento por meio do qual as mensagens são enviadas.

Para obter mais informações sobre as propriedades de um connection factory, consulte [“Propriedades de ConnectionFactory”](#) na página 182

## Sobre esta tarefa

Quaisquer espaços de fila ou de tópico que você precisa devem ser definidos. Por padrão, um espaço de tópico chamado Default.Topic.Space é definido durante a instalação do servidor, mas, se você precisar de espaços de tópico adicionais, deverá criar esses espaços de tópico você mesmo. Você não precisa predefinir tópicos individuais dentro de um espaço de tópico, uma vez que o servidor instancia esses tópicos individuais dinamicamente conforme necessário.

As etapas a seguir resumem o que você precisa fazer.

## Procedimento

1. Crie as filas que seu aplicativo precisa para o sistema de mensagens ponto a ponto.
2. Crie quaisquer espaços de tópico adicionais que seu aplicativo precise para o sistema de mensagens de publicação / assinatura.

## Resultados

Agora é possível iniciar seu aplicativo.

### Tarefas relacionadas

[Configurando o gerenciador de filas e o broker para um aplicativo que se conecta a um gerenciador de filas IBM MQ](#)

Esta seção assume que você está usando IBM WebSphere MQ 7.0.1 ou mais recente. Antes de poder executar um aplicativo que se conecta a um gerenciador de filas IBM MQ, você deve configurar o gerenciador de filas. Para um aplicativo de publicação / assinatura, alguma configuração adicional será necessária se você estiver usando a interface de publicação / assinatura enfileirada.

[Configurando um broker para um aplicativo que usa uma conexão em tempo real com um broker](#)  
Antes de poder executar um aplicativo que usa uma conexão em tempo real com um broker, você deve configurar esse broker.

## Usando os aplicativos de amostra XMS

---

Use os aplicativos de amostra fornecidos com o XMS para verificar sua instalação e configuração do servidor de sistema de mensagens e para ajudá-lo a construir seus aplicativos. As amostras fornecem uma visão geral dos recursos comuns de cada API.

### Conceitos relacionados

[Serviço da Web de Consulta de JNDI](#)

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus. Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

[Configurando o Ambiente do Servidor de Mensagens](#)

Os tópicos nesta seção descrevem como configurar o ambiente do servidor de sistema de mensagens para permitir que os aplicativos XMS se conectem a um servidor.

### Tarefas relacionadas

[Instalando Message Service Client for .NET usando o assistente de instalação](#)

A instalação usa um instalador MSI InstallShield X/Windows. Duas opções de configuração estão disponíveis, para que seja possível escolher uma instalação completa ou customizada.

## Os aplicativos de amostra:

Os aplicativos de amostra fornecem uma visão geral dos recursos comuns de cada API. É possível usá-los para verificar a sua instalação e o servidor de sistema de mensagens configurado e para ajudar a construir os seus próprios aplicativos.

Se você precisar de ajuda para criar seus próprios aplicativos, será possível usar os aplicativos de amostra como um ponto de início. Tanto a origem quanto uma versão compilada são fornecidas para cada aplicativo. Revise o código-fonte de amostra e identifique as etapas principais para criar cada objeto necessário para seu aplicativo (ConnectionFactory, Conexão, Sessão, Destino, e um Produtor, ou um Consumidor, ou ambos) e para configurar quaisquer propriedades específicas que sejam necessárias para especificar como você deseja que seu aplicativo funcione. Para obter mais informações, consulte “Gravando aplicativos do XMS” na página 20. As amostras estão sujeitas a mudanças em liberações futuras de XMS.

A tabela a seguir mostra os três conjuntos de aplicativos de amostra (um para cada API) fornecidos com o XMS.

Nome da amostra	Descrição
SampleConsumerCS	Um aplicativo consumidor de mensagens que obtém mensagens de uma fila ou assina um tópico.
SampleProducerCS	Um aplicativo de produtor de mensagem que produz mensagens para uma fila ou em um tópico.
SampleConfigCS	Um aplicativo de configuração que pode ser usado para criar um repositório de objeto administrado que se baseia em arquivo. O aplicativo contém um connection factory e um destino para suas configurações de conexão específicas. Esse repositório de objeto administrado pode, então, ser usado com cada um dos aplicativos de consumidor e de produtor de amostra.

As amostras que suportam as mesmas funções nas várias APIs possuem diferenças sintáticas.

- Os aplicativos de consumidor e de produtor de mensagens de amostra suportam as seguintes funções:
  - Conexões com IBM MQ, IBM Integration Bus (usando uma conexão em tempo real com um broker) e um WebSphere Application Server service integration bus
  - Consultas do Repositório de Objeto Administrado Usando a Interface de Contexto Inicial
  - Conexões com filas (IBM MQ e WebSphere Application Server service integration bus) e tópicos (IBM MQ, conexão em tempo real com um broker, e WebSphere Application Server service integration bus)
  - Mensagens de base, de byte, de mapa, de objeto, de fluxo e de texto
- O aplicativo consumidor de mensagens de amostra suporta modos de recebimento síncronos e assíncronos e instruções SQL Selector.
- O aplicativo do produtor de mensagem de amostra suporta modos de entrega persistentes e não persistentes.

## Modos de funcionamento

As amostras podem operar em um dos dois modos:

### Modo Simples

É possível executar as amostras com a entrada mínima do usuário.

### Modo Avançado

É possível customizar mais finamente a maneira na qual as amostras operam.

Todas as amostras são compatíveis e podem, portanto, operar por meio de linguagens.

### Conceitos relacionados

[Construindo seus próprios aplicativos](#)

Você constrói seus próprios aplicativos, como você constrói os aplicativos de amostra.

### Tarefas relacionadas

[Executando os Aplicativos de Amostra](#)

É possível executar os aplicativos de amostra .NET interativamente em modo simples ou avançado, ou não interativamente, usando arquivos de resposta gerados automaticamente ou gerados automaticamente.

#### Construindo os aplicativos de amostra .NET

Quando você constrói um aplicativo .NET de amostra, uma versão executável de sua amostra escolhida é criada.

## Executando os Aplicativos de Amostra

É possível executar os aplicativos de amostra .NET interativamente em modo simples ou avançado, ou não interativamente, usando arquivos de resposta gerados automaticamente ou gerados automaticamente.

### Antes de começar

Antes de executar qualquer um dos aplicativos de amostra fornecidos, você deve primeiro configurar o ambiente do servidor de sistema de mensagens para que os aplicativos possam se conectar a um servidor. Consulte [“Configurando o Ambiente do Servidor de Mensagens”](#) na página 13.

### Procedimento

Para executar um aplicativo de amostra .NET, conclua as etapas a seguir:

**Sugestão:** Quando você estiver executando um aplicativo de amostra, digite ? em qualquer momento para obter ajuda sobre o que fazer em seguida.

1. Selecione o modo no qual você deseja executar o aplicativo de amostra.

Digite `Advanced` ou `Simple`.

2. Responda às perguntas.

Para selecionar o valor padrão, que é mostrado entre colchetes no final da pergunta, pressione `Enter`. Para selecionar um valor diferente, digite o valor apropriado e pressione `Enter`.

Aqui está uma pergunta de exemplo:

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

Nesse caso, o valor padrão é `wpm` (conexão com um WebSphere Application Server service integration bus).

### Resultados

Quando você executa os aplicativos de amostra, os arquivos de resposta são gerados automaticamente no diretório de trabalho atual. Os nomes de arquivo de resposta estão no formato `connection_type-sample_type.rsp`; por exemplo, `wpm-producer.rsp`. Se necessário, é possível usar o arquivo de resposta gerado para executar novamente o aplicativo de amostra com as mesmas opções, de modo que não tenha que inserir as opções novamente.

### Conceitos relacionados

Os aplicativos de amostra:

Os aplicativos de amostra fornecem uma visão geral dos recursos comuns de cada API. É possível usá-los para verificar a sua instalação e o servidor de sistema de mensagens configurado e para ajudar a construir os seus próprios aplicativos.

### Tarefas relacionadas

Construindo os aplicativos de amostra .NET

Quando você constrói um aplicativo .NET de amostra, uma versão executável de sua amostra escolhida é criada.

## Construindo os aplicativos de amostra .NET

Quando você constrói um aplicativo .NET de amostra, uma versão executável de sua amostra escolhida é criada.

### Antes de começar

Instale o compilador apropriado. Essa tarefa supõe que o Microsoft Visual Studio 2012 esteja instalado e que você esteja familiarizado com sua utilização.

### Procedimento

Para construir um aplicativo de amostra .NET , conclua as etapas a seguir:

1. Clique no arquivo de solução `Samples.sln` fornecido com as amostras .NET.
2. Clique com o botão direito do mouse na solução `Amostras` na janela Explorador de Solução e selecione **Construir Solução**.

### Resultados

Um programa executável é criado na subpasta apropriada da amostra, seja `bin/Debug` ou `bin/Release`, dependendo da configuração que você escolheu. Esse programa possui o mesmo nome que a pasta, com um sufixo CS. Por exemplo, se você estiver construindo a versão C# do aplicativo de amostra do produtor de mensagens, `SampleProducerCS.exe` será criado na pasta `SampleProducer`.

### Conceitos relacionados

Os aplicativos de amostra:

Os aplicativos de amostra fornecem uma visão geral dos recursos comuns de cada API. É possível usá-los para verificar a sua instalação e o servidor de sistema de mensagens configurado e para ajudar a construir os seus próprios aplicativos.

“Construindo seus próprios aplicativos” na página 44

Você constrói seus próprios aplicativos, como você constrói os aplicativos de amostra.

### Tarefas relacionadas

Executando os Aplicativos de Amostra

É possível executar os aplicativos de amostra .NET interativamente em modo simples ou avançado, ou não interativamente, usando arquivos de resposta gerados automaticamente ou gerados automaticamente.

## Desenvolvendo aplicativos do XMS

---

Os tópicos nesta seção fornecem informações que podem ser úteis ao gravar aplicativos XMS .

Para obter informações sobre como gravar aplicativos XMS , consulte os tópicos a seguir:

### Gravando aplicativos do XMS

Os tópicos nesta seção fornecem informações para ajudá-lo ao gravar aplicativos XMS ..

Esta seção contém conceitos gerais para a gravação de aplicativos XMS. Consulte também “Gravando aplicativos XMS .NET” na página 44 para obter informações específicas para a criação de aplicativos .NET.

Esta seção contém os seguintes tópicos:

- “O modelo de encadeamento” na página 21
- “ConnectionFactories e objetos de Conexão” na página 21
- “Sessões” na página 24
- “Destinos” na página 29
- “Produtores de mensagens” na página 34
- “Consumidores de mensagens” na página 34

- [“Navegadores de fila” na página 38](#)
- [“Solicitantes” na página 38](#)
- [“Exclusão de objeto” na página 39](#)
- [“Tipos primitivos do XMS” na página 40](#)
- [“Conversão implícita de um valor de propriedade de um tipo de dados para outro” na página 41](#)
- [“Iteradores” na página 43](#)
- [“Identificadores de conjunto de caracteres codificados” na página 43](#)
- [“Códigos de erro e de exceção do XMS” na página 43](#)
- [“Construindo seus próprios aplicativos” na página 44](#)

### **Conceitos relacionados**

[Gravando aplicativos XMS .NET](#)

Os tópicos desta seção fornecem informações para ajudá-lo ao compor aplicativos XMS .NET.

### **Referências relacionadas**

[.REDE interfaces](#)

Esta seção documenta as interface de classe .REDE e suas propriedades e métodos.

## **O modelo de encadeamento**

As regras gerais controlam como um aplicativo multiencadeado pode usar objetos XMS .

- Apenas objetos dos tipos a seguir podem ser usados simultaneamente em encadeamentos diferentes:
  - ConnectionFactory
  - Conexão
  - ConnectionMetaData
  - Destino
- Um objeto de Sessão pode ser usado em apenas um único encadeamento em um determinado momento.

As exceções a essas regras são indicadas por entradas rotuladas "Contexto de encadeamento" nas definições de interface dos métodos em [“Clientes de serviço de mensagens para referência .NET” na página 91](#).

### **Conceitos relacionados**

[Condições de erro que podem ser manipuladas no tempo de execução](#)

Os códigos de retorno de chamadas de API são condições de erro que podem ser manipulados no tempo de execução. A maneira na qual você lida com esse tipo de erro depende se você está usando a API C ou C++.

## **ConnectionFactoryies e objetos de Conexão**

Um objeto ConnectionFactory fornece um modelo que um aplicativo usa para criar um objeto Connection. O aplicativo usa o objeto Connection para criar um objeto Session.

Para .NET, o aplicativo XMS usa primeiro um objeto XMSFactoryFactory para obter uma referência a um objeto ConnectionFactory que seja adequado ao tipo de protocolo necessário. Esse objeto ConnectionFactory pode, então, produzir conexões somente para esse tipo de protocolo.

Um aplicativo XMS pode criar várias conexões, e um aplicativo multiencadeado pode usar um único objeto de Conexão simultaneamente em vários encadeamentos. Um objeto Connection encapsula uma conexão de comunicações entre um aplicativo e um servidor de sistema de mensagens.

Uma conexão atende a vários propósitos:

- Quando um aplicativo cria uma conexão, o aplicativo pode ser autenticado.

- Um aplicativo pode associar um identificador de cliente exclusivo a uma conexão. O identificador de cliente é usado para suportar assinaturas duráveis no domínio de publicação / assinatura. O identificador de cliente pode ser configurado de duas maneiras:

A maneira preferida de designar um identificador de cliente de conexões é configurar em um objeto `ConnectionFactory` específico do cliente usando propriedades e designá-lo de forma transparente à conexão criada.

Uma maneira alternativa de designar um identificador de cliente é usar um valor específico do provedor que é configurado no objeto `Connection`. Esse valor não substitui o identificador que foi configurado administrativamente. Ele é fornecido para o caso em que não existe nenhum identificador administrativamente especificado. Se um identificador especificado administrativamente existir, uma tentativa de substituí-la com um valor específico do provedor fará com que uma exceção seja lançada. Se um aplicativo configurar explicitamente um identificador, ele deverá fazer isso imediatamente após a criação da conexão e antes de qualquer outra ação na conexão ser tomada; caso contrário, uma exceção será lançada.

Um aplicativo XMS normalmente cria uma conexão, uma ou mais sessões e um número de produtores de mensagens e consumidores de mensagens.

A criação de uma conexão é relativamente cara em termos de recursos do sistema, porque envolve o estabelecimento de uma conexão de comunicação e pode também envolver a autenticação do aplicativo.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto `ConnectionFactory` e `Destination` que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

#### `IConnectionFactory` (para a interface .NET)

Um aplicativo usa um `connection factory` para criar uma conexão.

#### Propriedades de `ConnectionFactory`

Uma visão geral das propriedades do objeto `ConnectionFactory`, com links para informações de referência mais detalhadas.

#### `IDestination` (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

#### Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### ***Modo iniciado e interrompido da conexão***

Uma conexão pode operar no modo iniciado ou interrompido.

Quando um aplicativo cria uma conexão, a conexão fica no modo interrompido. Quando a conexão está no modo interrompido, o aplicativo pode inicializar as sessões e pode enviar mensagens, mas não pode recebê-las, de forma síncrona ou assíncrona.

Um aplicativo pode iniciar uma conexão chamando o método `Start Connection`. Quando a conexão está no modo iniciado, o aplicativo pode enviar e receber mensagens. O aplicativo pode, então, parar e reiniciar a conexão chamando os métodos `Stop Connection` e `Start Connection`.

### **Conceitos relacionados**

#### Fechamento da conexão

Um aplicativo fecha uma conexão chamando o método `Fechar Conexão`.

#### Manipulação de Exceção

Se um aplicativo usar uma conexão apenas para consumir mensagens de forma assíncrona, ele aprenderá sobre um problema com a conexão somente usando um listener de exceções

### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

### **Fechamento da conexão**

Um aplicativo fecha uma conexão chamando o método Fechar Conexão.

Quando um aplicativo fecha uma conexão, XMS executa as ações a seguir:

- Ele fecha todas as sessões associadas à conexão e exclui determinados objetos associados a essas sessões. Para obter mais informações sobre quais objetos são excluídos, consulte [“Exclusão de objeto” na página 39](#). Ao mesmo tempo, XMS retrocede quaisquer transações atualmente em andamento dentro das sessões.
- Ele encerra a conexão de comunicações com o servidor de sistema de mensagens.
- Ele libera a memória e outros recursos internos usados pela conexão.

O XMS não reconhece o recebimento de nenhuma mensagem que ele tenha falhado em reconhecer durante uma sessão antes de fechar a conexão. Para obter mais informações sobre como reconhecer o recebimento de mensagens, consulte [“Reconhecimento de mensagem” na página 26](#).

### **Conceitos relacionados**

#### Modo iniciado e interrompido da conexão

Uma conexão pode operar no modo iniciado ou interrompido.

#### Manipulação de Exceção

Se um aplicativo usar uma conexão apenas para consumir mensagens de forma assíncrona, ele aprenderá sobre um problema com a conexão somente usando um listener de exceções

#### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

### **Manipulação de Exceção**

Se um aplicativo usar uma conexão apenas para consumir mensagens de forma assíncrona, ele aprenderá sobre um problema com a conexão somente usando um listener de exceções

As exceções XMS.NET são todas derivadas de System.Exception. Para obter mais informações, consulte [“Manipulação de erros no .NET” na página 48](#).

### **Conceitos relacionados**

#### Modo iniciado e interrompido da conexão

Uma conexão pode operar no modo iniciado ou interrompido.

#### Fechamento da conexão

Um aplicativo fecha uma conexão chamando o método Fechar Conexão.

#### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

### **Conexão com um barramento de integração de serviços**

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

O protocolo HTTP pode ser usado em situações em que uma conexão TCP/IP direta não é possível. Uma situação comum é quando se comunica através de um firewall, como quando duas empresas trocam mensagens. Usar HTTP para se comunicar por meio de um firewall geralmente é referido como *tunelamento HTTP*. O tunelamento HTTP, no entanto, é inerentemente mais lento do que o uso de uma conexão TCP/IP direta, porque os cabeçalhos de HTTP incluem significativamente a quantidade de dados que são transferidos e porque o protocolo HTTP requer mais fluxos de comunicação do que o TCP/IP.

Para criar uma conexão TCP/IP, um aplicativo pode usar um connection factory cuja propriedade [XMSC\\_WPM\\_TARGET\\_TRANSPORT\\_CHAIN](#) esteja configurada como

`XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC`. Esse é o valor padrão da propriedade. Se a conexão for criada com sucesso, a propriedade `XMSC_WPM_CONNECTION_PROTOCOL` da conexão será configurada como `XMSC_WPM_CP_TCP`.

Para criar uma conexão que utiliza HTTP, um aplicativo deve usar um connection factory cuja propriedade `XMSC_WPM_TARGET_TRANSPORT_CHAIN` esteja configurada para o nome de uma cadeia de transporte de entrada, que está configurada para usar um canal de transporte HTTP. Se a conexão for criada com sucesso, a propriedade `XMSC_WPM_CONNECTION_PROTOCOL` da conexão será configurada como `XMSC_WPM_CP_HTTP`. Para obter informações sobre como configurar cadeias de transporte, consulte [Configurando cadeias de transporte](#) na documentação do produto WebSphere Application Server .

Um aplicativo possui uma opção semelhante de protocolos de comunicação ao se conectar a um servidor de auto-inicialização. A propriedade `XMSC_WPM_PROVIDER_ENDPOINTS` de um connection factory é uma sequência de um ou mais endereços de terminal de servidores de autoinicialização. O componente de cadeia de transporte de auto-inicialização de cada endereço de terminal pode ser `XMSC_WPM_BOOTSTRAP_TCP`, para uma conexão TCP/IP com um servidor de auto-inicialização ou `XMSC_WPM_BOOTSTRAP_HTTP`, para uma conexão que usa HTTP.

### **Conceitos relacionados**

[Modo iniciado e interrompido da conexão](#)

Uma conexão pode operar no modo iniciado ou interrompido.

[Fechamento da conexão](#)

Um aplicativo fecha uma conexão chamando o método Fechar Conexão.

[Manipulação de Exceção](#)

Se um aplicativo usar uma conexão apenas para consumir mensagens de forma assíncrona, ele aprenderá sobre um problema com a conexão somente usando um listener de exceções

### **Tarefas relacionadas**

[Criando Objetos Administrados](#)

As definições de objeto `ConnectionFactory` e `Destination` que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

[IConnectionFactory \(para a interface .NET\)](#)

Um aplicativo usa um connection factory para criar uma conexão.

[Propriedades de ConnectionFactory](#)

Uma visão geral das propriedades do objeto `ConnectionFactory` , com links para informações de referência mais detalhadas.

[IDestination \(para a interface .NET\)](#)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

[Propriedades de Destino](#)

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

## **Sessões**

Uma sessão é um único contexto encadeado para enviar e receber mensagens.

Um aplicativo pode usar uma sessão para criar mensagens, produtores de mensagens, consumidores de mensagens, navegadores de filas e destinos temporários. Um aplicativo também pode usar uma sessão para executar transações locais.

Um aplicativo pode criar várias sessões, em que cada sessão produz e consome mensagens independentemente das outras sessões. Se dois consumidores de mensagem em sessões separadas (ou mesmo na mesma sessão) assinam o mesmo tópico, cada um recebe uma cópia de qualquer mensagem publicada nesse tópico.

Diferente de um objeto Connection, um objeto Session não pode ser usado simultaneamente em encadeamentos diferentes. Apenas o método Fechar Sessão de um objeto de Sessão pode ser chamado a partir de um encadeamento diferente daquele que o objeto Session está usando no momento. O método Fechar Sessão termina uma sessão e libera os recursos do sistema alocados para a sessão.

Se um aplicativo precisar processar mensagens simultaneamente em mais de um encadeamento, o aplicativo deverá criar uma sessão em cada encadeamento e, em seguida, usar essa sessão para qualquer operação de envio ou recebimento dentro desse encadeamento.

### **Sessões transacionadas**

Os aplicativos XMS podem executar transações locais. Uma *transação local* é aquela que envolve mudanças apenas para os recursos do gerenciador de filas ou do barramento de integração de serviços ao qual o aplicativo está conectado.

As informações neste tópico são relevantes apenas se um aplicativo se conectar a um gerenciador de filas IBM MQ ou a um barramento de integração de serviços WebSphere Application Server . As informações não são relevantes para uma conexão em tempo real com um broker.

Para executar transações locais, um aplicativo deve primeiro criar uma sessão transacionada ao chamar o método Create Session de um objeto Connection, especificando como um parâmetro que a sessão é transacionada. Em seguida, todas as mensagens enviadas e recebidas dentro da sessão são agrupadas em uma sequência de transações. Uma transação terminará quando o aplicativo confirmar ou recuperar as mensagens que ele enviou e recebeu desde o início da transação.

Para confirmar uma transação, um aplicativo chama o método Commit do objeto Session. Quando uma transação for confirmada, todas as mensagens enviadas dentro da transação se tornarão disponíveis para entrega para outros aplicativos e todas as mensagens recebidas dentro da transação serão confirmadas para que o servidor de sistema de mensagens não tente entregá-las ao aplicativo novamente. No domínio ponto a ponto, o servidor de sistema de mensagens também remove as mensagens recebidas a partir de suas filas.

Para retroceder uma transação, um aplicativo chama o método Rollback do objeto Session. Quando uma transação for retrocedida, todas as mensagens enviadas dentro da transação serão descartadas pelo servidor de sistema de mensagens e todas as mensagens recebidas dentro da transação se tornarão disponíveis para entrega novamente. No domínio de ponto a ponto, as mensagens que foram recebidas são colocadas de volta em suas filas e se tornam visíveis a outros aplicativos novamente.

Uma nova transação é iniciada automaticamente quando um aplicativo cria uma sessão transacionada ou chama o método Commit ou Rollback. Portanto, uma sessão transacionada sempre possui uma transação ativa.

Quando um aplicativo fechar uma sessão transacionada, um retrocesso implícito ocorrerá. Quando um aplicativo fechar uma conexão, um retrocesso implícito ocorrerá para todas as sessões transacionadas da conexão.

Uma transação é completamente contida dentro de uma sessão transacionada. Uma transação não pode abranger as sessões. Isso significa que não é possível para um aplicativo enviar e receber mensagens em duas ou mais sessões transacionadas e, em seguida, confirmar ou retroceder todas estas ações como uma única transação.

### **Conceitos relacionados**

#### Reconhecimento de mensagem

Cada sessão que não é transacionada possui um modo de confirmação que determina como as mensagens recebidas pelo aplicativo são confirmadas. Três modos de confirmação estão disponíveis e a opção de modo de confirmação afeta o design do aplicativo.

#### Entrega de Mensagem Assíncrona

XMS usa um encadeamento para manipular todas as entregas de mensagens assíncronas para uma sessão. Isso significa que apenas uma função de listener de mensagem ou um método onMessage () pode ser executado de cada vez.

#### Entrega de Mensagem Síncrona

As mensagens são entregues de forma síncrona para um aplicativo se o aplicativo usar os métodos `Receive` de `MessageConsumer` objetos.

#### Modo de entrega de mensagens

O XMS suporta dois modos de entrega de mensagens

### **Reconhecimento de mensagem**

Cada sessão que não é transacionada possui um modo de confirmação que determina como as mensagens recebidas pelo aplicativo são confirmadas. Três modos de confirmação estão disponíveis e a opção de modo de confirmação afeta o design do aplicativo.

As informações neste tópico são relevantes apenas se um aplicativo se conectar a um gerenciador de filas IBM MQ ou a um barramento de integração de serviços WebSphere Application Server . As informações não são relevantes para uma conexão em tempo real com um broker.

XMS usa o mesmo mecanismo para reconhecer o recebimento de mensagens que o JMS utiliza.

Se uma sessão não for transacionada, a maneira que as mensagens recebidas pelo aplicativo são confirmadas será determinada pelo modo de confirmação da sessão. Três modos de confirmação são descritos nos parágrafos a seguir:

#### **XMSC\_AUTO\_ACKNOWLEDGE**

A sessão confirma automaticamente cada mensagem recebida pelo aplicativo.

Se as mensagens forem entregues de forma síncrona para o aplicativo, a sessão confirmará o recebimento de uma mensagem toda vez que uma chamada `Receive` for concluída com sucesso.

Se o aplicativo receber uma mensagem com sucesso, mas uma falha evitar a ocorrência de confirmação, a mensagem se tornará disponível para a entrega novamente. O aplicativo deve, portanto, ser capaz de manipular uma mensagem que seja entregue novamente.

#### **XMSC\_DUPS\_OK\_ACKNOWLEDGE**

A sessão confirma as mensagens recebidas pelo aplicativo em momentos que ele seleciona.

O uso desse modo de confirmação reduz a quantidade de trabalho que a sessão deve fazer, mas uma falha que evita a confirmação de mensagens pode fazer com que mais de uma mensagem se torne disponível para entrega novamente. O aplicativo deve, portanto, ser capaz de manipular mensagens que são entregues novamente.

#### **XMSC\_CLIENT\_ACKNOWLEDGE**

O aplicativo confirma as mensagens que ele recebe chamando o método `Acknowledge` da classe `Message`.

O aplicativo pode confirmar o recebimento de cada mensagem individualmente ou receber um lote de mensagens e chamar o método `Acknowledge` apenas para a última mensagem que ele recebe. Quando o método `Acknowledge` for chamado, todas as mensagens recebidas desde a última vez que o método foi chamado serão confirmadas.

Juntamente com qualquer um destes modos de confirmação, um aplicativo pode interromper e reiniciar a entrega de mensagens em uma sessão chamando o método `Recover` da classe `Session`. Mensagens cujo recebimento tenha sido anteriormente não reconhecido são entregues novamente. No entanto, elas não podem ser entregues na mesma sequência em que foram entregues anteriormente. No entanto, mensagens de prioridade superior podem ter chegado e algumas das mensagens originais podem ter expirado. No domínio ponto a ponto, algumas das mensagens originais podem ter sido consumidas por outro aplicativo.

Um aplicativo pode determinar se uma mensagem está sendo entregue novamente examinando o conteúdo do campo de cabeçalho `JMSRedelivered` da mensagem. O aplicativo faz isso chamando o método `Get JMSRedelivered` da classe `Message`.

### **Conceitos relacionados**

Sessões transacionadas

Os aplicativos XMS podem executar transações locais. Uma *transação local* é aquela que envolve mudanças apenas para os recursos do gerenciador de filas ou do barramento de integração de serviços ao qual o aplicativo está conectado.

#### Entrega de Mensagem Assíncrona

XMS usa um encadeamento para manipular todas as entregas de mensagens assíncronas para uma sessão. Isso significa que apenas uma função de listener de mensagem ou um método `onMessage()` pode ser executado de cada vez.

#### Entrega de Mensagem Síncrona

As mensagens são entregues de forma síncrona para um aplicativo se o aplicativo usar os métodos `Receive` de `MessageConsumer` objetos.

#### Modo de entrega de mensagens

O XMS suporta dois modos de entrega de mensagens

### ***Entrega de Mensagem Assíncrona***

XMS usa um encadeamento para manipular todas as entregas de mensagens assíncronas para uma sessão. Isso significa que apenas uma função de listener de mensagem ou um método `onMessage()` pode ser executado de cada vez.

Se mais de um consumidor de mensagens em uma sessão estiver recebendo mensagens de forma assíncrona, e uma função de listener de mensagem ou um método `onMessage()` estiver entregando uma mensagem para um consumidor de mensagens, qualquer outro consumidor de mensagens que estiver aguardando a mesma mensagem deverá continuar aguardando. Outras mensagens que estão aguardando para serem entregues para a sessão também devem continuar aguardando.

Se um aplicativo requerer a entrega simultânea de mensagens, crie mais de uma sessão para que XMS use mais de um encadeamento para manipular a entrega de mensagens assíncronas. Dessa maneira, mais de uma função do listener de mensagem ou método `onMessage()` pode ser executado simultaneamente.

Uma sessão não é feita assíncrona, atribuindo um listener de mensagem a um consumidor. Uma sessão se torna assíncrona somente quando o método `Connection.Start` é chamado. Todas as chamadas síncronas são permitidas até que o método `Connection.Start` seja chamado. A entrega de mensagens para os consumidores iniciam quando o `Connection.Start` é chamado.

Se as chamadas síncronas, como a criação de um consumidor ou um produtor, devem ser feitas em uma sessão assíncrona, o `Connection.Stop` deve ser chamado. Uma sessão pode ser continuada chamando o método `Connection.Start` para iniciar a entrega de mensagens. A única exceção a isso é o encadeamento de entrega de mensagens da sessão, que é aquele que entrega mensagens para a função de retorno de chamada. Esse encadeamento pode fazer qualquer chamada na sessão (exceto uma chamada `Fechar`) na função de retorno de chamada de mensagem.

**Nota:** No modo Não gerenciado, a chamada `MQDISC` dentro de uma função de retorno de chamada não é suportada pelo cliente IBM MQ .NET. Portanto, o aplicativo cliente não pode Criar ou Fechar sessões dentro do retorno de chamada `MessageListener` no modo de recebimento Assíncrono. Crie e despose a sessão fora do método `MessageListener`.

### **Conceitos relacionados**

#### Sessões transacionadas

Os aplicativos XMS podem executar transações locais. Uma *transação local* é aquela que envolve mudanças apenas para os recursos do gerenciador de filas ou do barramento de integração de serviços ao qual o aplicativo está conectado.

#### Reconhecimento de mensagem

Cada sessão que não é transacionada possui um modo de confirmação que determina como as mensagens recebidas pelo aplicativo são confirmadas. Três modos de confirmação estão disponíveis e a opção de modo de confirmação afeta o design do aplicativo.

#### Entrega de Mensagem Síncrona

As mensagens são entregues de forma síncrona para um aplicativo se o aplicativo usar os métodos `Receive` de `MessageConsumer` objetos.

## Modo de entrega de mensagens

O XMS suporta dois modos de entrega de mensagens

### **Entrega de Mensagem Síncrona**

As mensagens são entregues de forma síncrona para um aplicativo se o aplicativo usar os métodos Receive de MessageConsumer objetos.

Usando os métodos Receive, um aplicativo pode esperar um período de tempo especificado para uma mensagem ou pode esperar indefinidamente. Como alternativa, se um aplicativo não desejar aguardar por uma mensagem, ele poderá usar o método Receive with No Wait.

#### **Conceitos relacionados**

##### Sessões transacionadas

Os aplicativos XMS podem executar transações locais. Uma *transação local* é aquela que envolve mudanças apenas para os recursos do gerenciador de filas ou do barramento de integração de serviços ao qual o aplicativo está conectado.

##### Reconhecimento de mensagem

Cada sessão que não é transacionada possui um modo de confirmação que determina como as mensagens recebidas pelo aplicativo são confirmadas. Três modos de confirmação estão disponíveis e a opção de modo de confirmação afeta o design do aplicativo.

##### Entrega de Mensagem Assíncrona

XMS usa um encadeamento para manipular todas as entregas de mensagens assíncronas para uma sessão. Isso significa que apenas uma função de listener de mensagem ou um método onMessage () pode ser executado de cada vez.

## Modo de entrega de mensagens

O XMS suporta dois modos de entrega de mensagens

### **Modo de entrega de mensagens**

O XMS suporta dois modos de entrega de mensagens

- Mensagens *persistentes* são entregues uma vez. Um servidor de sistema de mensagens toma precauções especiais, como a criação de log das mensagens, para assegurar que as mensagens persistentes não sejam perdidas em trânsito, mesmo no caso de uma falha.
- Mensagens *não persistentes* são entregues no máximo uma vez. Mensagens não persistentes são menos confiáveis do que as mensagens persistentes porque elas podem ser perdidas em trânsito no caso de uma falha.

A opção de modo de entrega é uma troca entre confiabilidade e desempenho. Mensagens não persistentes são geralmente transportadas mais rapidamente do que as mensagens persistentes.

#### **Conceitos relacionados**

##### Sessões transacionadas

Os aplicativos XMS podem executar transações locais. Uma *transação local* é aquela que envolve mudanças apenas para os recursos do gerenciador de filas ou do barramento de integração de serviços ao qual o aplicativo está conectado.

##### Reconhecimento de mensagem

Cada sessão que não é transacionada possui um modo de confirmação que determina como as mensagens recebidas pelo aplicativo são confirmadas. Três modos de confirmação estão disponíveis e a opção de modo de confirmação afeta o design do aplicativo.

##### Entrega de Mensagem Assíncrona

XMS usa um encadeamento para manipular todas as entregas de mensagens assíncronas para uma sessão. Isso significa que apenas uma função de listener de mensagem ou um método onMessage () pode ser executado de cada vez.

##### Entrega de Mensagem Síncrona

As mensagens são entregues de forma síncrona para um aplicativo se o aplicativo usar os métodos Receive de MessageConsumer objetos.

## Destinos

Um aplicativo XMS usa um objeto de Destino para especificar o destino das mensagens que estão sendo enviadas e a origem de mensagens que estão sendo recebidas.

Um aplicativo XMS pode criar um objeto Destination no tempo de execução ou obter um destino predefinido a partir do repositório de objetos administrados.

Como com um ConnectionFactory, a maneira mais flexível para um aplicativo XMS para especificar um destino é defini-la como um objeto administrado. Usando essa abordagem, os aplicativos gravados em linguagens C, C++ e .NET e Javapodem compartilhar definições do destino. As propriedades de objetos de Destino administrados podem ser mudadas sem alterar qualquer código.

Para aplicativos .NET, você cria um destino usando o método CreateTopic ou CreateQueue. Esses dois métodos estão disponíveis em ambos os objetos ISession e XMSFactoryFactory na API .NET. Para obter mais informações, consulte o [“Destinos em .NET”](#) na página 46 e o [“IDestination”](#) na página 110.

### Referências relacionadas

[IDestination \(para a interface .NET\)](#)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

[Propriedades de Destino](#)

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### Identificadores de recursos uniformes do tópico

O URI (Identificador Uniforme de Recursos (URI) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

O URI para um tópico inicia com o tópico de sequência: //, seguido pelo nome do tópico e (opcional) uma lista de pares nome-valor que configuram as propriedades do tópico restantes. Um nome de tópico não pode estar vazio.

Aqui está um exemplo em um fragmento de código .NET :

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

Para obter mais informações sobre as propriedades de um assunto, incluindo o nome e os valores válidos que podem ser usados em um URI, consulte [“Propriedades de Destino”](#) na página 189.

Ao especificar um URI de tópico para uso em uma assinatura, os curingas podem ser usados. A sintaxe para esses curingas depende do tipo de conexão e da versão do broker; as opções a seguir estão disponíveis:

- IBM WebSphere MQ 7.0 gerenciador de filas com formato de caractere curinga no nível de caractere
- IBM WebSphere MQ 7.0 gerenciador de filas com formato de curinga de nível de Tópico
- Barramento de integração de serviços WebSphere Application Server

### IBM WebSphere MQ 7.0 gerenciador de filas com formato de caractere curinga no nível de caractere

O gerenciador de filas IBM WebSphere MQ 7.0 com o formato de curinga no nível de Caractere usa os caracteres curinga a seguir:

- \* para 0 ou mais caracteres
- ? para 1 caractere
- % para um caractere de escape

[Tabela 1 na página 30](#) fornece alguns exemplos de como usar esse esquema curinga.

Tabela 1. URIs de exemplo usando esquema curinga no nível de caractere para o gerenciador de filas IBM WebSphere MQ 7.0

Identificador Uniforme de Recursos	Correspondentes	Examples
"topic://Sport *Results"	Todos os tópicos iniciados com "Sport" e terminam em "Results"	"topic://SportsResults" e "topic://Sport/Hockey/National/Div3/Results"
"topic://Sport?Results"	Todos os tópicos iniciando com "Sport" seguido por um caractere único, seguido por "Resultados"	"topic://SportsResults" e "topic://SportXResults"
"topic://Sport/ * ball*/Div? / Results/*/??"	tópicos	"topic://Sport/Football/Div1/Results/2002/Nov" e "topic://Sport/Netball/National/Div3/Results/02/Jan"

## IBM WebSphere MQ 7.0 gerenciador de filas com formato de curinga de nível de Tópico

O gerenciador de filas IBM WebSphere MQ 7.0 com o formato de curinga de nível de Tópico usa os seguintes caracteres curinga:

- # para corresponder a diversos níveis
- + para corresponder a um único nível

Tabela 2 na página 30 fornece alguns exemplos de como usar esse esquema curinga.

Tabela 2. URIs de exemplo usando esquema curinga no nível de tópico para o gerenciador de filas IBM WebSphere MQ 7.0

Identificador Uniforme de Recursos	Correspondentes	Examples
"topic://Sport/ + / Results"	Todos os Tópicos com um Nome de Nível Hierárquico Único entre Esporte e Resultados	"topic://Sport/Football/Results" e "topic://Sport/Ju-Jitsu/Results"
"topic://Sport/#/Results"	Todos os tópicos iniciando com "Sport/" e terminando em "/Results"	"topic://Sport/Football/Results" e "topic://Sport/Hockey/National/Div3/Results"
" topic://Sport/ Football/#"	Todos os tópicos iniciando com "Sport/ Football/"	"topic://Sport/Football/Results" e "topic://Sport/Football/TeamNews/Signings/Managerial"

## Barramento de integração de serviços WebSphere Application Server

O barramento de integração de serviços WebSphere Application Server usa os seguintes caracteres curinga:

- \* para corresponder a qualquer caractere em um nível na hierarquia
- // para corresponder a 0 ou mais níveis
- //. para combinar com 0 ou mais níveis (no final de uma expressão Tópico)

Tabela 3 na página 31 fornece alguns exemplos de como usar esse esquema curinga.

Tabela 3. URIs de exemplo usando esquema curinga para o barramento de integração de serviços WebSphere Application Server

Identificador Uniforme de Recursos	Correspondentes	Examples
"topic://Sport/ * ball/ Results"	Todos os tópicos com um nome de nível hierárquico único terminando em "ball" entre Esporte e Resultados	"topic://Sport/Football/Results" e "topic://Sport/Netball/Results"
"topic://Sport// Results"	Todos os tópicos iniciando com "Sport/" e terminando em "/Results"	"topic://Sport/Football/Results" e "topic://Sport/Hockey/National/Div3/Results"
"topic://Sport/ Football//."	Todos os tópicos iniciando com "Sport/ Football/"	"topic://Sport/Football/Results" e "topic://Sport/Football/TeamNews/ Signings/Managerial"
"topic://Sport/ * ball// Results//."	tópicos	"topic://Sport/Football/Results" e "topic://Sport/Netball/National/Div3/ Results/2002/November"

### Conceitos relacionados

Identificadores uniformes de recursos da fila

O URI para uma fila específica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

Destinos Temporários

Os aplicativos XMS podem criar e usar destinos temporários.

Curinga de destino

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

### Referências relacionadas

IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### **Identificadores uniformes de recursos da fila**

O URI para uma fila específica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

O URI para uma fila começa com a fila de sequência: //, seguido pelo nome da fila; ele também pode incluir uma lista de pares nome-valor que configuram as propriedades da fila restantes..

Para as filas IBM MQ (mas não para filas do provedor de mensagens padrão WebSphere Application Server), o gerenciador de filas no qual a fila reside pode ser especificado antes da fila, com um / separando o nome do gerenciador de filas do nome da fila.

Se um gerenciador de filas for especificado, ele deverá ser aquele para o qual o XMS está diretamente conectado para a conexão usando essa fila ou deve ser acessível a partir desta fila. Os gerenciadores de filas remotas são suportados apenas para recuperar mensagens das filas, não para colocar mensagens nas filas. Para obter detalhes completos, consulte a documentação do gerenciador de filas IBM MQ .

Se nenhum gerenciador de filas for especificado, então, o extra / separador será opcional e sua presença ou ausência não fará diferença para a definição da fila.

As definições de fila a seguir são equivalentes para uma fila do IBM MQ chamada QB em um gerenciador de filas chamado QM\_A, à qual o XMS é conectado diretamente:

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

### **Conceitos relacionados**

#### Identificadores de recursos uniformes do tópico

O URI (Identificador Uniforme de Recursos (URI) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

#### Destinos Temporários

Os aplicativos XMS podem criar e usar destinos temporários.

#### Curinga de destino

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

### **Referências relacionadas**

#### IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

#### Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### ***Destinos Temporários***

Os aplicativos XMS podem criar e usar destinos temporários.

Um aplicativo geralmente usa um destino temporário para receber respostas para solicitar mensagens. Para especificar o destino no qual uma resposta a uma mensagem de solicitação deve ser enviada, um aplicativo chama o método `Set JMSReplyTo` do objeto de mensagem que representa a mensagem de solicitação. O destino especificado na chamada pode ser um destino temporário.

Embora uma sessão seja usada para criar um destino temporário, o escopo de um destino temporário é, na verdade, a conexão que foi usada para criar a sessão. Qualquer uma das sessões da conexão pode criar produtores de mensagens e consumidores de mensagens para o destino temporário. O destino temporário permanece até que seja explicitamente excluído ou que a conexão termine, o que ocorrer primeiro.

Quando um aplicativo cria uma fila temporária, uma fila é criada no servidor de sistema de mensagens para o qual o aplicativo está conectado. Se o aplicativo estiver conectado a um gerenciador de filas, uma fila dinâmica será criada a partir da fila modelo cujo nome é especificado pela propriedade `XMSC_WMQ_TEMPORARY_MODEL` e o prefixo usado para formar o nome da fila dinâmica será especificado pela propriedade `XMSC_WMQ_TEMP_Q_PREFIX`. Se o aplicativo estiver conectado a um barramento de integração de serviços, uma fila temporária será criada no barramento e o prefixo usado para formar o nome da fila temporária será especificado pela propriedade `XMSC_WPM_TEMP_Q_PREFIX`.

Quando um aplicativo que está conectado a um barramento de integração de serviços cria um tópico temporário, o prefixo usado para formar o nome do tópico temporário é especificado pela propriedade `XMSC_WPM_TEMP_TOPIC_PREFIX`.

### **Conceitos relacionados**

#### Identificadores de recursos uniformes do tópico

O URI (Identificador Uniforme de Recursos (URI) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

#### Identificadores uniformes de recursos da fila

O URI para uma fila especifica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

### Curinga de destino

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

### **Referências relacionadas**

IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### ***Curinga de destino***

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

Os esquemas são:

<b>Tipo de conexão</b>	<b>Esquema curinga</b>	<b>Descrição</b>
WebSphere MQ gerenciador de filas	*	0 ou mais caracteres
	?	1 caractere
	%	Caractere de escape
Conexão de tempo real com um broker	#	Corresponder vários níveis
	+	Corresponder um único nível
WebSphere Serviço Integration Bus	*	Corresponder quaisquer caracteres em um nível na hierarquia
	//	Corresponder 0 ou mais níveis
	//.	Corresponder 0 ou mais níveis (no final de uma expressão de Tópico)

### **Conceitos relacionados**

Identificadores de recursos uniformes do tópico

O URI (Identificador Uniforme de Recursos (URI) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

Identificadores uniformes de recursos da fila

O URI para uma fila especifica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

Destinos Temporários

Os aplicativos XMS podem criar e usar destinos temporários.

### **Referências relacionadas**

IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

## Produtores de mensagens

Em XMS, um produtor de mensagens pode ser criado com um destino válido ou sem destino associado. Ao criar um produtor de mensagem com um destino nulo, um destino válido precisa ser especificado ao enviar uma mensagem.

### ***Produtores de mensagens sem destino associado***

Em XMS.NET, um produtor de mensagens pode ser criado com um destino nulo.

Para criar um produtor de mensagem sem destino associado ao usar a API .NET, NULL deve ser transmitido como um parâmetro no método `CreateProducer()` do objeto `ISession` (por exemplo, `session.CreateProducer(null)`). No entanto, um destino válido deve ser especificado quando a mensagem for enviada

### ***Produtores de mensagens com destino associado***

Nesse cenário, o produtor da mensagem é criado usando um destino válido. Durante a operação de envio, o destino não precisa ser especificado.

## Consumidores de mensagens

Os consumidores de mensagens podem ser classificados como assinantes duráveis e não duráveis e consumidores de mensagens síncrona e assíncrona.

### ***Assinantes duráveis***

Um assinante durável é um consumidor de mensagens que recebe todas as mensagens publicadas em um tópico, incluindo mensagens publicadas enquanto o assinante está inativo.

As informações neste tópico são relevantes apenas se um aplicativo se conectar a um gerenciador de filas IBM MQ ou a um barramento de integração de serviços WebSphere Application Server. As informações não são relevantes para uma conexão em tempo real com um broker.

Para criar um assinante durável para um tópico, um aplicativo chama o método `Criar Assinante Durável` de um objeto de Sessão, especificando como parâmetros um nome que identifica a assinatura durável e um objeto de Destino que representa o tópico. O aplicativo pode criar um assinante durável com ou sem um seletor de mensagem, e pode especificar se o assinante durável deve receber mensagens publicadas por sua própria conexão.

A sessão usada para criar um assinante durável deve ter um identificador de cliente associado. O identificador de cliente é o mesmo que aquele associado à conexão que é usada para criar a sessão; ela é especificada conforme descrito em [“ConnectionFactories e objetos de Conexão” na página 21](#).

O nome que identifica a assinatura durável deve ser exclusivo dentro do identificador de cliente e, portanto, o identificador de cliente faz parte do identificador completo e exclusivo da assinatura durável. O servidor de sistema de mensagens mantém um registro da assinatura durável e assegura que todas as mensagens publicadas no tópico sejam retidas até que sejam reconhecidas pelo assinante durável ou que elas expirem.

O servidor de sistema de mensagens continua a manter o registro da assinatura durável mesmo depois que o assinante durável for fechado. Para reutilizar uma assinatura durável que foi criada anteriormente, um aplicativo deve criar um assinante durável especificando o mesmo nome de assinatura e usando uma sessão com o mesmo identificador de cliente, como aqueles associados à assinatura durável. Apenas uma sessão de cada vez pode ter um assinante permanente para uma assinatura durável específica.

O escopo de uma assinatura durável é o servidor de sistema de mensagens que está mantendo um registro da assinatura. Se dois aplicativos conectados a diferentes servidores de sistema de mensagens criarem um assinante durável usando o mesmo nome de assinatura e o identificador de cliente, duas assinaturas duráveis completamente independentes serão criadas.

Para excluir uma assinatura durável, um aplicativo chama o método `Unsubscribe` de um objeto `Session`, especificando como um parâmetro o nome que identifica a assinatura durável. O identificador de cliente associado à sessão deve ser o mesmo que aquele associado à assinatura durável. O servidor de

mensagens exclui o registro da assinatura durável que ele está mantendo e não envia mais mensagens para o assinante durável.

Para alterar uma assinatura existente, um aplicativo pode criar um assinante durável usando o mesmo nome de assinatura e identificador de cliente, mas especificando um tópico diferente ou seletor de mensagem (ou ambos). Alterar uma assinatura durável é equivalente a excluir a assinatura e criar uma nova.

Para um aplicativo que se conecta ao IBM WebSphere MQ 7.0 gerenciador de filas, XMS gerencia as filas de assinantes. Portanto, o aplicativo não é necessário para especificar uma fila de assinantes. O XMS ignorará a fila de assinantes, se especificada.

No entanto, para um aplicativo que se conecta ao gerenciador de filas do IBM WebSphere MQ 6.0, cada assinante durável deve ter uma fila de assinantes designada. Para especificar o nome da fila de assinantes para um tópico, configure a propriedade `XMSC_WMQ_DUR_SUBQ` do objeto de Destino que representa o tópico. A fila de assinantes padrão é `SYSTEM.JMS.D.SUBSCRIBER.QUEUE`.

Os assinantes duráveis que se conectam aos gerenciadores de fila do IBM WebSphere MQ 6.0 podem compartilhar uma única fila de assinantes ou cada assinante durável pode recuperar suas próprias mensagens de sua própria fila de assinantes exclusivos. Para obter uma discussão sobre qual abordagem adotar para seu aplicativo, consulte *XMS IBM WebSphere MQ Usando Java*.

Observe que não é possível alterar a fila de assinantes para uma assinatura durável. A única maneira de alterar a fila de assinantes é excluir a assinatura e criar uma nova.

Para um aplicativo que se conecta a um barramento de integração de serviços, cada assinante durável deve ter um nome de assinatura durável designado. Para especificar o início da assinatura durável para todos os assinantes permanentes que usam a mesma conexão, configure a propriedade `XMSC_WPM_DUR_SUB_HOME` do objeto `ConnectionFactory` usado para criar a conexão. Para especificar o início da assinatura durável para um tópico individual, configure a propriedade `XMSC_WPM_DUR_SUB_HOME` do objeto `Destination` que representa o tópico. Um lar de assinaturas duráveis deve ser especificado para uma conexão antes que um aplicativo possa criar um assinante durável que use a conexão. Qualquer valor especificado para um destino substitui o valor especificado para a conexão.

### **Assinantes Não Duráveis**

Um assinante não durável é um consumidor de mensagens que recebe apenas mensagens que são publicadas enquanto o assinante está ativo. As mensagens entregues enquanto o assinante está inativo são perdidas.

As informações neste tópico são relevantes apenas quando você está usando o sistema de mensagens de publicar / assinar sobre um gerenciador de filas IBM WebSphere MQ 6.0.

Se os objetos do consumidor não forem excluídos antes ou durante o fechamento da conexão, as mensagens poderão permanecer nas filas do broker para assinantes que não estão mais ativos.

Nessa situação, as mensagens podem ser limpas nessas filas usando o utilitário de Limpeza fornecido com o IBM WebSphere MQ classes for JMS Classes for JMS. Os detalhes de como usar esse utilitário são fornecidos em *IBM WebSphere MQ Usando Java*. Você também pode precisar aumentar a profundidade da fila da fila de assinantes se houver grandes números de mensagens restantes nessa fila.

### **Consumidores de mensagens síncrona**

O consumidor de mensagens síncronas recebe as mensagens de uma fila de forma síncrona

Um consumidor de mensagens síncrona recebe uma mensagem de cada vez. Quando o método `Receive(wait interval)` é usado; a chamada aguarda apenas um período de tempo especificado em milissegundos para uma mensagem ou até o consumidor de mensagens ser fechado.

Se o método `ReceiveNoWait()` for usado, o consumidor de mensagens síncrona receberá mensagens sem nenhum atraso; se a próxima mensagem estiver disponível, ela será recebida imediatamente, caso contrário, um ponteiro para um objeto de Mensagem nulo será retornado.

## **Consumidores de mensagens assíncronas**

O consumidor de mensagens assíncronas recebe uma mensagem de uma fila de forma assíncrona. O listener de mensagem registrado pelo aplicativo é chamado sempre que uma nova mensagem está disponível na fila.

## **Mensagens suspeitas no XMS**

Uma mensagem suspeita é aquela que não pode ser processada por um aplicativo MDB de recebimento. Se uma mensagem suspeita for encontrada, o objeto XMS MessageConsumer poderá reenqueueá-la de acordo com duas propriedades de fila, `BOQUEUE` e `BOTHRESH`.

Em algumas circunstâncias, uma mensagem entregue a um MDB pode ser retrocedida em uma fila do IBM MQ. Isso pode acontecer, por exemplo, quando uma mensagem é entregue dentro de uma unidade de trabalho que é, subsequentemente, retrocedida. Uma mensagem que é retrocedida é geralmente entregue novamente, mas uma mensagem mal formatada pode repetidamente fazer com que um MDB falhe e, portanto, não pode ser entregue. Essa mensagem é chamada de uma mensagem suspeita. É possível configurar IBM MQ para que a mensagem suspeita seja automaticamente transferida para outra fila para investigação adicional ou seja descartada. Para obter informações sobre como configurar o IBM MQ desta maneira, consulte [Manipulando mensagens suspeitas no ASF](#).

Às vezes, uma mensagem mal formatada incorretamente chega em uma fila. Nesse contexto, mal formatada significa que o aplicativo de recebimento não pode processar a mensagem corretamente. Essa mensagem pode fazer com que o aplicativo de recebimento falhe e restaure essa mensagem mal formatada. A mensagem pode então ser entregue repetidamente à fila de entrada e recuperada repetidamente pelo aplicativo. Essas mensagens são conhecidas como mensagens suspeitas. O objeto XMS MessageConsumer detecta mensagens suspeitas e roteia-as para um destino alternativo.

O gerenciador de filas IBM MQ mantém um registro do número de vezes que cada mensagem foi restaurada. Quando esse número atinge um valor limite configurável, o consumidor de mensagem recoloca a mensagem em uma fila de restauração denominada. Se esse novo enfileiramento falhar por qualquer razão, a mensagem será removida da fila de entrada e um enfileirada novamente na fila de mensagens não entregues ou descartada.

Os objetos de `ConnectionConsumer XMS` manipulam mensagens suspeitas da mesma maneira e usando as mesmas propriedades de fila. Se diversos consumidores de conexão estiverem monitorando a mesma fila, é possível que a mensagem suspeita possa ser entregue a um aplicativo mais vezes do que o valor limite antes que o novo enfileiramento ocorra. Este comportamento ocorre devido à maneira como consumidores de conexões individuais monitoram filas e enfileiram mensagens suspeitas novamente.

O valor do limite e o nome da fila de restauração são atributos de uma fila do IBM MQ. Os nomes dos atributos são `BackoutThreshold` e `BackoutRequeueQName`. A fila à qual eles se aplicam é a seguinte:

- Para o sistema de mensagens ponto a ponto, é a fila local subjacente. Isso é importante quando os consumidores de mensagens e os consumidores de conexão usam aliases de filas.
- Para o sistema de mensagens de publicação/assinatura no modo normal do provedor do sistemas de mensagens do IBM MQ, ela é a fila modelo por meio da qual a fila gerenciada do tópico é criada.
- Para o sistema de mensagens de publicar/assinar no modo de migração do provedor do sistema de mensagens do IBM MQ, é a fila `CCSUB` definida no objeto `TopicConnectionFactory` ou a fila `CCDSUB` definida no objeto `Topic`.

Para configurar os atributos `BackoutThreshold` e `BackoutRequeueQName`, emita o comando `MQSC` a seguir:

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

Para o sistema de mensagens de publicação / assinatura, se o seu sistema criar uma fila dinâmica para cada assinatura, esses valores de atributo serão obtidos das classes WebSphere MQ para a fila modelo JMS, SYSTEM.JMS.MODEL.QUEUE. Para alterar essas configurações, use:

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

Se o valor do limite de restauração for zero, a manipulação de mensagens suspeitas será desativada e as mensagens suspeitas permanecerão na fila de entrada. Caso contrário, quando a contagem de restaurações atingir o valor do limite, a mensagem será enviada para a fila de restauração denominada.

Se a contagem de restauração atingir o valor do limite, mas a mensagem não puder ir para a fila de restauração, a mensagem será enviada para a fila de mensagens não entregues ou, se a mensagem for não persistente, ela será descartada.

Essa situação ocorre se a fila de restauração não estiver definida ou se o objeto MessageConsumer não puder enviar a mensagem para a fila de restauração.

## Configurando o seu sistema para executar a manipulação de mensagens suspeitas

A fila que o XMS .NET usa ao consultar os atributos **BOTHRESH** e **BOQNAME** depende do estilo de sistema de mensagens que está sendo executado:

- Para o sistema de mensagens ponto a ponto, é a fila local subjacente. Isso é importante quando um aplicativo XMS .NET está consumindo mensagens de filas de alias ou de filas de clusters.
- Para o sistema de mensagens de publicação/assinatura, uma fila gerenciada é criada para conter as mensagens para um aplicativo. O XMS .NET consulta a fila gerenciada para determinar os valores para os atributos **BOTHRESH** e **BOQNAME**.

A fila gerenciada é criada por meio de uma fila modelo associada ao objeto Tópico que o aplicativo assinou e herda os valores dos atributos **BOTHRESH** e **BOQNAME** da fila modelo. A fila modelo que é usada depende de o aplicativo de recebimento ter obtido uma assinatura durável ou não durável:

- A fila modelo usada para assinaturas duráveis é especificada pelo atributo **MDURMDL** do Tópico. O valor padrão desse atributo é SYSTEM.DURABLE.MODEL.QUEUE.
- Para assinaturas não duráveis, a fila modelo que é usada é especificada pelo atributo **MNDURMDL**. O valor padrão do atributo **MNDURMDL** é SYSTEM.NDURABLE.MODEL.QUEUE.

Ao consultar os atributos **BOTHRESH** e **BOQNAME**, o XMS .NET:

- Abre a fila local ou a fila de destino para uma fila de alias.
- Consulta os atributos **BOTHRESH** e **BOQNAME**.
- Fecha a fila local ou a fila de destino para uma fila de alias.

As opções de abertura usadas ao abrir uma fila local ou a fila de destino para uma fila de alias, dependem da versão do IBM MQ que está sendo usada:

- Para IBM MQ 9.0.0 Fix Pack 8e anterior, se a fila local, ou a fila de destino para uma fila de alias, for uma fila de clusters, então XMS .NET abre a fila com as opções MQ00\_INPUT\_AS\_Q\_DEF, MQ00\_INQUIRE e MQ00\_FAIL\_IF QUIESCING. Isso significa que o usuário que estiver executando o aplicativo de recebimento deverá ter acesso de consulta e de obtenção à instância local da fila de clusters.

O XMS .NET abre todos os outros tipos de fila local com as opções de abertura MQ00\_INQUIRE e MQ00\_FAIL\_IF QUIESCING. Para que o XMS .NET consulte os valores dos atributos, o usuário que executa o aplicativo de recebimento deve ter acesso de consulta na fila local.

- **V 9.0.0.9** Ao usar XMS .NET de IBM MQ 9.0.0 Fix Pack 9, o usuário que está executando o aplicativo de recebimento deve ter acesso de consulta na fila local, independentemente do tipo da fila.

Para mover mensagens suspeitas para uma fila de reenfileiramento de restauração ou para a fila de mensagens não entregues do gerenciador de filas, deve-se conceder ao usuário que estiver executando o aplicativo as autoridades put e passall.

### *Manipulando mensagens suspeitas no ASF*

Ao usar o Application Server Facilities (ASF), o `ConnectionConsumer`, em vez de o `MessageConsumer`, processa mensagens suspeitas. O `ConnectionConsumer` recoloca mensagens na fila de acordo com as propriedades `BackoutThreshold` e `BackoutRequeueQName` da fila.

Quando um aplicativo usa `ConnectionConsumers`, as circunstâncias nas quais uma mensagem é restaurada dependem da sessão que o servidor de aplicativos fornece:

- Quando a sessão é não transacionada, com `AUTO_ACKNOWLEDGE` ou `DUPS_OK_ACKNOWLEDGE`, uma mensagem é restaurada somente após um erro do sistema ou se o aplicativo for finalizado inesperadamente.
- Quando a sessão não é transacionada com `CLIENT_RECONHEÇO`, mensagens não reconhecidas podem ser restauradas pelo servidor de aplicativos que chama `Session.recover()`.

Geralmente, a implementação do cliente de `MessageListener` ou as chamadas do servidor de aplicativos `Message.acknowledge()` `Message.acknowledge()` reconhece todas as mensagens entregues na sessão até agora.

- Quando a sessão é transacionada, mensagens não reconhecidas podem ser restauradas pelo servidor de aplicativos chamando `Session.rollback()`.

## **Navegadores de fila**

Um aplicativo usa um navegador de filas para pesquisar mensagens em uma fila sem removê-las.

Para criar um navegador de filas, um aplicativo chama o método `Criar Navegador de Filas` de um objeto de `ISession`, especificando como um parâmetro um objeto de `Destino` que identifica a fila a ser procurada. O aplicativo pode criar um navegador de filas com ou sem um seletor de mensagem.

Depois de criar um navegador de filas, o aplicativo pode chamar o método `GetEnumerator` do objeto `IQueueBrowser` para obter uma lista de mensagens na fila. O método retorna um enumerador que encapsula uma lista de objetos de `Mensagem`. A ordem dos objetos de `Mensagem` na lista é a mesma que a ordem em que as mensagens seriam recuperadas da fila. O aplicativo pode então usar o enumerador para pesquisar cada mensagem por vez.

O enumerador é atualizado dinamicamente conforme as mensagens são colocadas na fila e removidas da fila. Cada vez que o aplicativo chama `IEnumerator.MoveNext()` para navegar na próxima mensagem na fila, a mensagem reflete os conteúdos atuais da fila.

Um aplicativo pode chamar o método `GetEnumerator` mais de uma vez para um navegador de filas fornecido. Cada chamada retorna um novo enumerador. O aplicativo pode, portanto, usar mais de um enumerador para procurar as mensagens em uma fila e manter várias posições dentro da fila.

Um aplicativo pode usar um navegador de filas para procurar por uma mensagem adequada para remover de uma fila e, em seguida, usar um consumidor de mensagens com um seletor de mensagem para remover a mensagem. O seletor de mensagem pode selecionar a mensagem de acordo com o valor do campo de cabeçalho `JMSMessageID`. Para obter informações sobre esse e outros campos de cabeçalho da mensagem JMS, consulte [“Campos de cabeçalho em uma mensagem XMS” na página 71](#).

## **Solicitantes**

Um aplicativo usa um solicitante para enviar uma mensagem de solicitação e, em seguida, esperar e receber a resposta.

Muitos aplicativos do sistema de mensagens são baseados em algoritmos que enviam uma mensagem de solicitação e, em seguida, aguardam uma resposta. XMS fornece uma classe chamada `Requestor` para ajudar com o desenvolvimento desse estilo de aplicativo.

Para criar um solicitante, um aplicativo chama o construtor `Create Requestor` da classe `Requestor`, especificando como parâmetros um objeto `Session` e um objeto de `Destino` que identifica onde as mensagens de solicitação devem ser enviadas. A sessão não deve ser transacionada nem ter um modo de confirmação de `XMSC_CLIENT_ACKNOWLEDGE`. O construtor cria automaticamente uma fila ou um tópico provisório para os quais as mensagens de resposta devem ser enviadas.

Depois de criar um solicitante, o aplicativo pode chamar o método Request do objeto Solicitante para enviar uma mensagem de solicitação e, em seguida, aguardar e receber uma resposta do aplicativo que recebe a mensagem de solicitação. A chamada aguarda até que a resposta seja recebida ou até que a sessão termine, o que ocorrer primeiro. Somente uma resposta é requerida pelo solicitante para cada mensagem de solicitação.

Quando o aplicativo fecha o solicitante, a fila temporária ou o tópico é excluído. A sessão associada, no entanto, não fecha.

## Exclusão de objeto

Quando um aplicativo exclui um objeto XMS que ele criou, XMS libera os recursos internos que foram alocados para o objeto.

Quando um aplicativo cria um objeto do XMS, o XMS aloca memória e outros recursos internos para o objeto. XMS retém esses recursos internos até que o aplicativo exclua explicitamente o objeto chamando o método close ou delete do objeto, no ponto em que o XMS libera os recursos internos. Se um aplicativo tentar excluir um objeto que já está excluído, a chamada será ignorada.

Quando um aplicativo exclui um objeto Connection ou Session, o XMS exclui determinados objetos associados automaticamente e libera seus recursos internos. Esses são objetos que foram criados pelo objeto Connection ou Session e não têm função independente do objeto. Esses objetos são mostrados em Tabela 4 na página 39.

**Nota:** Se um aplicativo fechar uma conexão com sessões dependentes, todos os objetos dependentes dessas sessões também serão excluídos. Apenas um objeto Connection ou Session pode ter objetos dependentes.

<i>Tabela 4. Objetos que são excluídos automaticamente</i>		
<b>Objeto excluído</b>	<b>Método</b>	<b>Objetos Dependentes que São Excluídos Automaticamente</b>
Conexão	Encerrar Conexão	Objetos ConnectionMetaData e Session
Session	Fechar Sessão	Objetos MessageConsumer, MessageProducer, QueueBrowser e Requestor

## Transações XA IBM MQ Gerenciadas por meio de XMS

Transações XA IBM MQ gerenciadas podem ser usadas por meio de XMS.

Para usar as transações XA por meio do XMS, uma sessão transacionada deve ser criada. Quando a transação XA está em uso, o controle de transação é por meio de transações globais do Distributed Transaction Coordinator (DTC) e não é por meio de sessões do XMS. Ao usar transações XA, Session.commit ou Session.rollback não pode ser emitido na sessão XMS. Em vez disso, use os métodos DTC Transscope.Commit ou Transscope.Rollback consolidando ou retroceda as transações. Se uma sessão for usada para transação XA, o produtor ou o consumidor que são criados usando a sessão deve ser uma parte da transação XA. Eles não podem ser usados para nenhuma operação fora do escopo de transação XA. Eles não podem ser usados para operações como Producer.send ou Consumer.receive fora da transação XA.

Um objeto de exceção IllegalStateException será lançado se

- A sessão transacionada por XA é usada para Session.commit ou Session.rollback.
- Os objetos do produtor ou do consumidor que são usados uma vez na sessão transacionada XA são usados fora do escopo de transação XA.

As transações XA não são suportadas em consumidores assíncronos.

### Nota:

1. Um fechamento pode ser emitido no objeto Producer, Consumer, Session, ou Connection antes da confirmação da transação XA. Em quais casos, as mensagens na transação são retrocedidas. Da mesma forma, se a conexão for interrompida antes da confirmação da transação XA, todas as

mensagens na transação serão retrocedidas. Para um objeto `Producer`, um retrocesso significa que as mensagens não são colocadas na fila. Para um objeto `Consumer`, uma recuperação significa que as mensagens permanecem na fila.

2. Se um objeto `Producer` colocar uma mensagem com `TimeToLive` no `TransactionScope` e `commit` for emitido após o tempo decorrido, a mensagem poderá expirar antes que o `commit` seja emitido. Nesse caso, a mensagem não é disponibilizada para os objetos `Consumer`
3. Objetos `Session` não são suportados em encadeamentos. O uso de transações com objetos `Session` que são compartilhados entre encadeamentos não é suportado.

## Tipos primitivos do XMS

O XMS fornece equivalentes dos oito tipos primitivos Java (`byte`, `short`, `int`, `long`, `float`, `double`, `char` e `boolean`). Isso permite a troca de mensagens entre o XMS e o JMS sem que os dados se tornem perdidos ou corrompidos.

Tabela 5 na página 40 lista o tipo de dados equivalentes Java, o tamanho e o valor mínimo e máximo de cada tipo primitivo XMS.

*Tabela 5. Tipos de Dados XMS e seus Equivalentes Java*

Tipo de Dados XMS	Tipo de Dados Java Compatível	Tamanho	Valor mínimo	Valor máximo
System.Boolean	booleano	32 bits	false	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	short	16 bits	$-2^{15}$ (-32768)	$2^{15}-1$ (32767)
System.Int32	int	32 bits	$-2^{31}$ (-2147483648)	$2^{31}-1$ (2147483647)
System.Int64	grande	64 bits	$-2^{63}$ (-9223372036854775808)	$2^{63}-1$ (9223372036854775807)
System.Single	float	32 bits	$-3.402823E-38$ (para precisão de 7 dígitos)	$3.402823E + 38$ (para precisão de 7 dígitos)
System.Double	duplo	64 bits	$-1.79769313486231E-308$ (para 15-dígitos de precisão)	$1.79769313486231E + 308$ (para precisão de 15 dígitos)

### Conceitos relacionados

#### Atributos e Propriedades de Objetos

Um objeto XMS pode ter atributos e propriedades, que são características do objeto, que são implementados de diferentes maneiras

#### Conversão implícita de um valor de propriedade de um tipo de dados para outro

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

### Referências relacionadas

#### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

## Conversão implícita de um valor de propriedade de um tipo de dados para outro

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

Uma propriedade de um objeto tem um nome e um valor; o valor tem um tipo de dado associado, em que o valor de uma propriedade também é referido como o tipo de propriedade ..

Um aplicativo usa os métodos da classe PropertyContext para obter e configurar as propriedades de objetos. Para obter o valor de uma propriedade, um aplicativo chama o método que é apropriado para o tipo de propriedade. Por exemplo, para obter o valor de uma propriedade de número inteiro, um aplicativo geralmente chama o método GetIntProperty.

No entanto, quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Por exemplo, para obter o valor de uma propriedade de número inteiro, um aplicativo pode chamar o método GetStringProperty, que retorna o valor da propriedade como uma sequência. As conversões suportadas pelo XMS são mostradas em [Tabela 6 na página 41](#).

<b>Tipo de propriedade</b>	<b>Tipos de dados de destino suportados</b>
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
Matriz System.SByte	System.String
System.Int16	System.String, System.Int32, System.Int64

As regras gerais a seguir controlam as conversões suportadas:

- Os valores de propriedade numérica podem ser convertidos de um tipo de dados para outro, desde que nenhum dado seja perdido durante a conversão. Por exemplo, o valor de uma propriedade com o tipo de dados System.Int32 pode ser convertido em um valor com o tipo de dados System.Int64, mas não pode ser convertido em um valor com tipo de dados System.Int16.
- Um valor de propriedade de qualquer tipo de dados pode ser convertido em uma sequência.
- Um valor de propriedade de sequência pode ser convertido para qualquer outro tipo de dados, desde que a sequência seja formatada corretamente para a conversão. Se um aplicativo tentar converter um valor de propriedade de sequência que não está formatado corretamente, XMS poderá retornar erros.
- Se um aplicativo tentar uma conversão não suportada, o XMS poderá retornar um erro.

As regras a seguir se aplicam quando um valor de propriedade é convertido de um tipo de dados para outro:

- Ao converter um valor de propriedade booleana para uma sequência, o valor true é convertido para a sequência "true" e o valor false é convertido para a sequência "false".
- Ao converter um valor de propriedade booleana para um tipo de dados numérico, incluindo System.SByte, o valor true será convertido para 1 e o valor false será convertido para 0.

- Ao converter um valor de propriedade de sequência para um valor booleano, a sequência "true" (não faz distinção entre maiúsculas e minúsculas) ou "1" é convertida para true e a sequência "false" (sem distinção entre maiúsculas e minúsculas) ou "0" é convertida para false. Todas as outras sequências não podem ser convertidas.
- Ao converter um valor de propriedade de sequência para um valor com tipo de dados System.Int32, System.Int64, System.SByte ou System.Int16, a sequência deve ter o formato a seguir:

[ *espaços em branco* ] [ *sinal* ] *dígitos*

Os componentes de sequência são definidos da seguinte forma:

***espaços em branco***

Caracteres em branco à esquerda opcionais.

***sinal***

Um caractere de sinal de mais (+) ou de sinal de menos (-) opcional.

***dígitos***

Uma sequência contígua de caracteres de dígito (0-9). Pelo menos um caractere de dígito deve estar presente.

Após a sequência de caracteres de dígito, a sequência pode conter outros caracteres que não são caracteres de dígito, mas a conversão para assim que o primeiro desses caracteres for atingido. A sequência é assumida para representar um número inteiro decimal.

XMS pode retornar um erro se a sequência não estiver formatada corretamente.

- Ao converter um valor de propriedade de sequência para um valor com tipo de dados System.Double ou System.Float, a sequência deve ter o formato a seguir:

[ *blanks* ] [ *sign* ] [ *digits* ] [ *point*[ *d\_digits* ] ] [ *e\_char*[ *e\_sign* ] *e\_digits* ]

Os componentes de sequência são definidos da seguinte forma:

***espaços em branco***

(Opcional) Leitura de caracteres em branco.

***sinal***

(Opcional) Sinal de mais (+) ou sinal de menos (-).

***dígitos***

Uma sequência contígua de caracteres de dígito (0-9). Pelo menos um caractere de dígito deve estar presente em *digits* ou *d\_digits*.

***ponto***

(Opcional) Separador decimal (.).

***d\_digits***

Uma sequência contígua de caracteres de dígito (0-9). Pelo menos um caractere de dígito deve estar presente em *digits* ou *d\_digits*.

***e\_char***

Um caractere expoente, que é um *E* ou *e*.

***e\_sign***

(Opcional) Sinal de mais (+) ou sinal de menos (-) para o expoente.

***e\_digits***

Uma sequência contígua de caracteres de dígito (0-9) para o expoente. Pelo menos um caractere de dígito deve estar presente se a sequência contiver um caractere expoente.

Após a sequência de caracteres de dígito ou os caracteres opcionais que representam um expoente, a sequência pode conter outros caracteres que não são caracteres de dígito, mas a conversão para assim que o primeiro desses caracteres for atingido. Supõe-se que a sequência represente um número de vírgula flutuante decimal com um expoente que é uma potência de 10.

XMS pode retornar um erro se a sequência não estiver formatada corretamente.

- Ao converter um valor de propriedade numérico em uma sequência, incluindo um valor de propriedade com tipo de dados System.SByte, o valor é convertido para a representação de sequência do valor como

um número decimal, não a sequência que contém o caractere ASCII para esse valor. Por exemplo, o número inteiro 65 será convertido para a sequência "65", não para a sequência "A".

- Ao converter um valor de propriedade matriz de bytes em uma sequência, cada byte é convertido nos 2 caracteres hexadecimais que representam o byte. Por exemplo, a matriz de bytes {0}xF1, 0x12, 0x00, 0xFF } é convertida para a sequência "F11200FF".

Conversões de um tipo de propriedade para outros tipos de dados são suportadas pelos métodos de ambas as classes Property e PropertyContext.

### **Conceitos relacionados**

#### Atributos e Propriedades de Objetos

Um objeto XMS pode ter atributos e propriedades, que são características do objeto, que são implementados de diferentes maneiras

#### Tipos primitivos do XMS

O XMS fornece equivalentes dos oito tipos primitivos Java (byte, short, int, long, float, double, char e boolean). Isso permite a troca de mensagens entre o XMS e o JMS sem que os dados se tornem perdidos ou corrompidos.

### **Referências relacionadas**

#### Mensagens de Mapa

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

#### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

## **Iteradores**

Um agente iterativo encapsula uma lista de objetos e um cursor que mantém a posição atual na lista. O conceito de um Iterador, conforme disponível em IBM Message Service Client for C/C++ , é implementado usando a interface IEnumerator em IBM Message Service Client for .NET.

Quando um agente iterativo é criado, a posição do cursor é anterior ao primeiro objeto. Um aplicativo usa um agente iterativo para recuperar cada objeto por sua vez.

A classe Iterator de IBM Message Service Client for C/C++ é equivalente à classe Enumerator em Java.XMS O .NET é semelhante ao Java e usa uma interface IEnumerator.

Um aplicativo pode usar um IEnumerator para executar as tarefas a seguir:

- Para obter as propriedades de uma mensagem
- Para obter os pares nome-valor no corpo de uma mensagem de mapa
- Para procurar as mensagens em uma fila
- Para obter os nomes das propriedades de mensagem definidas pelo JMS suportadas por uma conexão

## **Identificadores de conjunto de caracteres codificados**

No XMS .NET, todas as sequências são transmitidas usando a sequência .NET nativa. Como isso tem uma codificação fixa, nenhuma informação adicional é necessária para interpretá-la. Portanto, a propriedade XMSC\_CLIENT\_CCID não é necessária para aplicativos XMS .NET .

## **Códigos de erro e de exceção do XMS**

XMS usa um intervalo de códigos de erro para indicar falhas. Esses códigos de erro não são explicitamente listados nesta documentação porque eles podem variar de liberação para liberação.

Apenas códigos de exceção XMS (no formato XMS\_X\_...) são documentados porque permanecem os mesmos em diferentes versões de XMS.

## Construindo seus próprios aplicativos

Você constrói seus próprios aplicativos, como você constrói os aplicativos de amostra.

Construa seu aplicativo .NET , conforme descrito em [“Construindo os aplicativos de amostra .NET”](#) na página 20, que também lista os pré-requisitos necessários para construir seus próprios aplicativos .NET . Para obter orientação adicional sobre como construir seus próprios aplicativos, use os makefiles fornecidos para cada aplicativo de amostra.

**Sugestão:** Para ajudar no diagnóstico de problemas no caso de uma falha, é possível achar útil compilar aplicativos com símbolos incluídos.

### Conceitos relacionados

[Os aplicativos de amostra:](#)

Os aplicativos de amostra fornecem uma visão geral dos recursos comuns de cada API. É possível usá-los para verificar a sua instalação e o servidor de sistema de mensagens configurado e para ajudar a construir os seus próprios aplicativos.

### Referências relacionadas

[.REDE interfaces](#)

Esta seção documenta as interface de classe .REDE e suas propriedades e métodos.

[Propriedades de objetos XMS](#)

Isso capítulo documenta as propriedades de objeto definidas por XMS

### **IBM MQ Reconexão de cliente automática por meio de XMS**

Configure o cliente XMS para reconectar automaticamente após uma falha de rede, gerenciador de filas ou servidor ao usar o cliente IBM WebSphere MQ 7.1 e mais recente, como o provedor de mensagens.

Use as propriedades WMQ\_CONNECTION\_NAME\_LIST e WMQ\_CLIENT\_RECONNECT\_OPTIONS da classe MQConnectionFactory para configurar uma conexão do cliente para reconectar automaticamente. A reconexão de cliente automática reconecta um cliente após uma falha de conexão ou como uma opção após parar o gerenciador de filas. O design de alguns aplicativos clientes os torna inadequados para reconexão automática.

As conexões do cliente reconectáveis automaticamente tornam-se reconectáveis depois que a conexão for estabelecida.

**Nota:** As propriedades Opções de Reconexão de Cliente, Tempo Limite de Reconexão de Clientee Lista de Nomes de Conexão também podem ser configuradas por meio da Tabela de Definições de Canais de Cliente (CCDT) ou ativando a reconexão do cliente por meio do arquivo mqclient.ini .

**Nota:** Se as propriedades de reconexão forem configuradas no objeto ConnectionFactory e, assim como na CCDT, a regra de precedência será a seguinte. Se o valor padrão da propriedade da lista de nomes de conexão for configurado no objeto ConnectionFactory , então, a CCDT terá precedência. Se a lista de nomes de conexão não estiver configurada para seu valor padrão, os valores da propriedade definidos no objeto ConnectionFactory têm precedência. O valor padrão da lista de nomes de conexão é localhost(1414).

## Gravando aplicativos XMS .NET

Os tópicos desta seção fornecem informações para ajudá-lo ao compor aplicativos XMS .NET.

Esta seção fornece informações que são específicas para a composição de aplicativos XMS .NET. Para obter informações gerais sobre a gravação de aplicativos XMS , consulte [“Gravando aplicativos do XMS”](#) na página 20.

Esta seção contém os seguintes tópicos:

- [“Tipos de dados para .NET”](#) na página 45

- [“Operações gerenciadas e não gerenciadas no .NET” na página 46](#)
- [“Destinos em .NET” na página 46](#)
- [“Propriedades em .NET” na página 47](#)
- [“Manipulação de Propriedades Não Existentes no .NET” na página 48](#)
- [“Manipulação de erros no .NET” na página 48](#)
- [“Listeners de mensagem e de exceção em .NET” na página 49](#)

### Conceitos relacionados

[Gravando aplicativos do XMS](#)

Os tópicos nesta seção fornecem informações para ajudá-lo ao gravar aplicativos XMS ..

### Referências relacionadas

[.REDE interfaces](#)

Esta seção documenta as interface de classe .REDE e suas propriedades e métodos.

## Tipos de dados para .NET

O XMS.NET suporta System.Boolean, System.Byte, System.SByte, System.Char, System.String, System.Single, System.Double, System.Decimal, System.Int32, System.Int64, System.UInt16, System.UInt32, System.UInt64 e System.Object. Os tipos de dados para XMS .NET são diferentes dos tipos de dados para C/C++ XMS. Você pode utilizar este tópico para identificar os tipos de dados correspondentes.

A tabela a seguir mostra os tipos de dados XMS .NET e XMS C/C++ correspondentes e os descreve resumidamente.

<i>Tabela 7. Tipos de dados para XMS.NET e XMS C/C++</i>		
<b>Tipo de XMS .NET</b>	<b>XMS tipo de C/C++</b>	<b>Descrição</b>
System.SByte	xmsSBYTE xmsINT8	Valor de 8 bits sinalizado
System.Byte	xmsBYTE xmsUINT8	Valor de 8 bits não assinado
System.Int16	xmsINT16 xmsSHORT	Valor de 16 bits sinalizado
System.UInt16	xmsUINT16 xmsUSHORT	Valor de 16 bits não assinado
System.Int32	xmsINT32 xmsINT	Valor de 32 bits assinado
System.UInt32	xmsUINT32 xmsUINT	Valor de 32 bits não assinado
System.Int64	xmsLONG xmsINT64	Valor de 64 Bits Assinado
System.UInt64	xmsULONG xmsUINT64	Valor de 64 bits não assinado

Tabela 7. Tipos de dados para XMS.NET e XMS C/C++ (continuação)

Tipo de XMS .NET	XMS tipo de C/C++	Descrição
System.Char	xmsCHAR16	Caractere de 16 bits não assinado (Unicode para .NET)
System.Single	xmsFLOAT	IEEE 32-bit float
System.Double	xmsDOUBLE	IEEE 64-bit float
System.Boolean	xmsBOOL	Um valor True / False
Não-aplicável	xmsCHAR	Valor de 8-bit assinado ou não assinado (depende da plataforma)
System.Decimal	Não-aplicável	Número inteiro sinalizado de 96 bits 10 <sup>0</sup> a 10 <sup>28</sup>
System.Object	Não-aplicável	Base de todos os tipos
System.String	Não-aplicável	Tipo de sequência

## Operações gerenciadas e não gerenciadas no .NET

O código gerenciado é executado exclusivamente dentro do ambiente Common Language Runtime do .NET e depende totalmente dos serviços fornecidos por esse tempo de execução. Um aplicativo é classificado como não gerenciado se qualquer parte do aplicativo executar ou chamar serviços fora do ambiente de tempo de execução de linguagem comum .NET .

Determinadas funcionalidades avançadas não podem ser suportadas atualmente dentro do ambiente gerenciado do .NET.

Se o seu aplicativo requer alguma funcionalidade que não é atualmente suportado no ambiente totalmente gerenciado, é possível alterar seu aplicativo para usar o ambiente não gerenciado sem requerer uma mudança substancial no seu aplicativo. No entanto, é necessário observar que a pilha XMS faz uso de código não gerenciado quando essa seleção é feita.

## Conexões com um gerenciador de filas IBM MQ

Conexões gerenciadas com WMQ\_CM\_CLIENT não suportará comunicações não TCP e compactação de canal. No entanto, essas conexões podem ser suportadas pelo uso de uma conexão não gerenciada (WMQ\_CM\_CLIENT\_UNMANAGED). Para obter mais informações, veja [Desenvolvendo aplicativos .NET](#).

Se você criar um connection factory a partir de um objeto administrado em um ambiente não gerenciado, você deverá alterar manualmente o valor para o modo de conexão para XMSC\_WMQ\_CM\_CLIENT\_UNMANAGED.

## Conexões com um mecanismo de mensagens de barramento de integração de serviços WebSphere Application Server

Conexões com um mecanismo de sistema de mensagens do barramento de integração de serviços WebSphere Application Server que requerem o uso do protocolo SSL (incluindo HTTPS) não são atualmente suportadas como código gerenciado.

### Referências relacionadas

[XMSC\\_WMQ\\_CONNECTION\\_MODE](#)

## Destinos em .NET

Em .NET, os destinos são criados de acordo com o tipo de protocolo e podem ser usados apenas no tipo de protocolo para o qual eles são criados.

Duas funções são fornecidas para criar destinos, um para tópicos e um para filas:

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

Essas funções estão disponíveis nos dois objetos a seguir na API:

- `ISession`
- `XMSFactoryFactory`

Em ambos os casos, esses métodos podem aceitar uma sequência de estilo de URI, que pode incluir parâmetros, no formato a seguir:

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

Como alternativa, esses métodos podem aceitar um nome de destino apenas, ou seja, um nome sem um tópico: `//` ou `queue://` prefix e sem parâmetros.

Isso significa que a sequência de estilo de URI a seguir:

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

produziria o mesmo resultado que o nome de destino a seguir:

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

Como para o barramento de integração de serviços JMS do WebSphere Application Server, os tópicos também podem ser especificados em um formato abreviado, que inclui o *topicname* e o *topicspace*, mas não pode incluir parâmetros:

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

## Propriedades em .NET

Um aplicativo .NET usa os métodos na interface `PropertyContext` para obter e configurar as propriedades de objetos.

A interface `PropertyContext` encapsula métodos que obtêm e configuram propriedades. Esses métodos são herdados, direta ou indiretamente, pelas classes a seguir:

- [BytesMessage](#)
- [Conexão](#)
- [ConnectionFactory](#)
- [ConnectionMetaData](#)
- [Destino](#)
- [MapMessage](#)
- [Mensagem](#)
- [MessageConsumer](#)
- [MessageProducer](#)
- [ObjectMessage](#)
- [QueueBrowser](#)
- [Sessão](#)
- [StreamMessage](#)
- [TextMessage](#)

Se um aplicativo configurar o valor de uma propriedade, o novo valor substituirá qualquer valor anterior que a propriedade possuiu.

Para obter mais informações sobre propriedades XMS , consulte [“Propriedades de objetos XMS” na página 181.](#)

Para facilidade de uso, os nomes e valores da propriedade XMS em XMS são predefinidos como constantes públicas em uma estrutura chamada XMSC. Os nomes dessas constantes estão no formato `XMSC.constant`; por exemplo, `XMSC.USERID` (uma constante de nome de propriedade) e `XMSC.DELIVERY_AS_APP` (uma constante de valor).

Além disso, é possível acessar constantes IBM MQ usando o `IBM.XMS.MQC struct`. Se o namespace `IBM.XMS` já estiver importado, será possível acessar os valores para essas propriedades no formato `MQC.constant`. Por exemplo, `MQC.MQRO_COA_WITH_FULL_DATA`.

Além disso, se você tiver um aplicativo híbrido que use as classes XMS .NET e IBM MQ para .NET e que importe ambos os espaços de nomes `IBM.XMS` e `IBM.WMQ`, então você deve qualificar totalmente o espaço de nomes de estrutura MQC para garantir que cada ocorrência seja única.

Algumas funcionalidades avançadas não são atualmente suportadas dentro do ambiente gerenciado .NET . Consulte [“Operações gerenciadas e não gerenciadas no .NET” na página 46](#) para obter mais detalhes.

## Manipulação de Propriedades Não Existentes no .NET

O manuseio de propriedades inexistentes no XMS .NET é amplamente consistente com a especificação JMS, e também com as implementações C e C++ de XMS.

No JMS, o acesso a uma propriedade não existente pode resultar em uma exceção do sistema Java quando um método tenta converter o valor não existente (nulo) para o tipo necessário. Se uma propriedade não existir, ocorrerão as seguintes exceções:

- `getStringProperty` e `getObjectProperty` return null
- `getBooleanProperty` retorna false porque `Boolean.valueOf (null)` retorna false
- `getIntProperty` etc. throw `java.lang.NumberFormatException` porque `Integer.valueOf (null)` throws a exceção

Se uma propriedade não existir no XMS .NET, ocorrerão as exceções a seguir:

- `GetStringProperty` e `GetObjectProperty` (e `GetBytesProperty`) retornam nulo (que é o mesmo que Java)
- `GetBooleanProperty` throws `System.NullReferenceException`
- `GetIntProperty` etc. throws `System.NullReferenceException`

Essa implementação é diferente de Java, mas ela é amplamente consistente com a especificação JMS, e com as interfaces C e C++ XMS. Assim como a implementação Java, XMS .NET propaga quaisquer exceções da chamada `System.Convert` para o responsável pela chamada. Ao contrário de Java, no entanto, XMS lança explicitamente `NullReferenceExceptions` em vez de apenas usar o comportamento nativo da estrutura .NET através de passagem de nulo para rotinas de conversão do sistema. Se seu aplicativo configurar uma propriedade para uma Cadeia como "abc" e chamar `GetIntProperty`, o `System.FormatException` lançado pelo `Convert.ToInt32 ("abc")` será propagado para o responsável pela chamada, o que é consistente com Java. `MessageFormatException` será lançada apenas se os tipos usados para `setProperty` e `getProperty` forem incompatíveis. Esse comportamento também é consistente com Java.

## Manipulação de erros no .NET

Exceções XMS .NET são todas derivadas de `System.Exception`. Chamadas de método XMS podem lançar exceções XMS específicas como `MessageFormatException`, gerais `XMSExceptions` ou exceções de sistema como `NullReferenceException`.

Grave aplicativos para capturar qualquer um desses erros, em blocos de captura específicos ou em blocos de captura gerais `System.Exception` , conforme apropriado para os requisitos de aplicativos.

## Listeners de mensagem e de exceção em .NET

Um aplicativo .NET usa um listener de mensagem para receber mensagens de forma assíncrona, e ele usa um listener de exceção para ser notificado de forma assíncrona de um problema com uma conexão.

A funcionalidade de ambos os listeners de mensagem e exceção é a mesma para .NET e para C++. No entanto, existem algumas pequenas diferenças de implementação.

## Listeners de mensagens no .REDE

Para receber mensagens de forma assíncrona, você deve concluir as seguintes etapas:

1. Defina um método que corresponda à assinatura do delegado de listener de mensagem. O método que você define pode ser um método estático ou de instância e pode ser definido em qualquer classe acessível. A assinatura do delegado é a seguinte:

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

e, portanto, você poderia definir o método como:

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

2. Instancie este método como um delegado usando algo semelhante ao seguinte:

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

3. Registre o delegado com um ou mais consumidores, configurando-o para a propriedade MessageListener do consumidor:

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

É possível remover o delegado, reconfigurando o MessageListener para nulo:

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

## Listeners de exceção no .NET

O listener de exceção funciona da mesma maneira que o listener de mensagem, mas possui uma definição de delegação diferente e é designado para a conexão, em vez disso, o consumidor de mensagens. Isso é o mesmo que para C++.

1. Defina o método. A assinatura do delegado é a seguinte:

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

e, portanto, o método definido poderia ser:

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

2. Instancie esse método como um delegado usando algo semelhante a:

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

3. Registre o delegado com a conexão configurando sua propriedade ExceptionListener:

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

É possível remover o delegado, reconfigurando o `ExceptionListener` para:

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

Quando nenhuma referência a eles permanecer, as exceções ou mensagens serão excluídas automaticamente pelo coletor de lixo do sistema

A seguir está um código de amostra.:

```
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);
    }
}
```

## Trabalhando com Objetos Administrados

Os tópicos nesta seção fornecem informações sobre objetos administrados. Os aplicativos XMS podem recuperar as definições de objeto de um repositório de objetos administrados centrais e usá-los para criar connection factories e destinos.

Esta seção fornece informações para ajudar na criação e gerenciamento de objetos administrados, descrevendo os tipos de repositório de objeto administrado que o XMS suporta. A seção também explica como um aplicativo XMS faz uma conexão com um repositório de objetos administrados para recuperar os objetos administrados necessários.

Esta seção contém os seguintes tópicos:

- [“Tipos Suportados de Repositório de Objetos Administrados”](#) na página 51
- [“Mapeamento de Propriedades para Objetos Administrados”](#) na página 52
- [“Propriedades necessárias para objetos ConnectionFactory administrados”](#) na página 55
- [“Propriedades Necessárias para Objetos de Destino Administrados”](#) na página 57
- [“Criando Objetos Administrados”](#) na página 58
- [“Objetos InitialContext”](#) na página 60
- [“Propriedades InitialContext”](#) na página 61
- [“Formato de URI para contextos iniciais XMS”](#) na página 62
- [“Serviço da Web de Consulta de JNDI”](#) na página 64
- [“Recuperação de Objetos Administrados”](#) na página 66

### **Conceitos relacionados**

#### Objetos Administrados

Usando objetos administrados, é possível administrar as configurações de conexão usadas pelos aplicativos clientes a serem administradas a partir de um repositório central. Um aplicativo recupera definições de objeto do repositório central e as utiliza para criar objetos `ConnectionFactory` e `Destination`. Usando objetos administrados, é possível desacoplar aplicativos dos recursos usados no tempo de execução.

## **Tipos Suportados de Repositório de Objetos Administrados**

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

Diretórios de objetos do Sistema de Arquivos assumem a forma de objetos serializados Java Naming Directory Interface (JNDI). Diretórios de objetos LDAP são diretórios que contêm objetos JNDI. Os diretórios de objetos do Sistema de Arquivos e LDAP podem ser administrados usando a ferramenta JMSAdmin, que é fornecida com IBM WebSphere MQ 6.0, ou o IBM MQ Explorer, que é fornecido com IBM WebSphere MQ 7.0 e mais recente. Tanto o sistema de Arquivos como os diretórios de objetos LDAP podem ser usados para administrar conexões de clientes, centralizando fábricas de conexão IBM WebSphere MQ e destinos. O administrador de rede pode implementar vários aplicativos que se referem ao mesmo repositório central e que são automaticamente atualizados para refletir as mudanças nas configurações de conexão feitas no repositório central.

Um CORBA Naming Directory contém fábricas de conexão WebSphere Application Server service integration bus e destinos e pode ser administrado usando o console administrativo WebSphere Application Server. Para que um aplicativo XMS recupere objetos do CORBA Naming Directory, um serviço da web de consulta JNDI deve ser implementado. Esse serviço da web não está disponível em todos os WebSphere Application Server service integration technologies. Consulte a documentação do produto para obter detalhes.

**Nota:** Reinicie as conexões do aplicativo para que as mudanças no diretório de objeto tenham efeito.

### **Conceitos relacionados**

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Propriedades InitialContext

Os parâmetros do construtor `InitialContext` incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

#### Objetos Administrados

Usando objetos administrados, é possível administrar as configurações de conexão usadas pelos aplicativos clientes a serem administradas a partir de um repositório central. Um aplicativo recupera definições de objeto do repositório central e as utiliza para criar objetos ConnectionFactory e Destination . Usando objetos administrados, é possível desacoplar aplicativos dos recursos usados no tempo de execução.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

#### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

## **Mapeamento de Propriedades para Objetos Administrados**

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

Por exemplo, para criar uma connection factory do XMS com as propriedades recuperadas de uma connection factory de JMS do IBM MQ, as propriedades devem ser mapeadas entre as duas.

Todos os mapeamentos de propriedade são executados automaticamente.

A tabela a seguir demonstra os mapeamentos entre algumas das propriedades mais comuns de connection factories e destinos. As propriedades mostradas nesta tabela são apenas um pequeno conjunto de exemplos, e nem todas as propriedades mostradas são relevantes para todos os tipos de conexão e servidores.

Tabela 8. Exemplos de mapeamento de nome para connection factory e propriedades de destino

Nome da propriedade IBM MQ JMS	Nome da propriedade XMS	Nome da propriedade WebSphere Application Server service integration bus
PERSISTÊNCIA (POR)	<u>XMSC_DELIVERY_MODE</u>	
EXPIRAÇÃO (EXP)	<u>XMSC_TIME_TO_LIVE</u>	
PRIORIDADE (PRI)	<u>XMSC_PRIORITY</u>	
	<u>XMSC_WPM_HOST_NAME</u>	serverName
	<u>XMSC_WPM_BUS_NAME</u>	busName
	<u>XMSC_WPM_TOPIC_SPACE</u>	topicName

Tabela 9. Propriedades do XMS .NET

Propriedade	Tipo de Objeto				
	CF	QCF	TCF	Fila	Tópico
<u>APPLICATIONNAME</u>	Y	Y	Y	N/D	N/D
<u>ASYNCEXCEPTION</u>	Y	Y	Y	N/D	N/D
<u>CCDTURL</u>	Y	Y	Y	N/D	N/D
<u>CHANNEL</u>	Y	Y	Y	N/D	N/D
<u>CONNECTIONNAMELIST</u>	Y	Y	Y	N/D	N/D
<u>CLIENTRECONNECTOPTIONS</u>	Y	Y	Y	N/D	N/D
<u>CLIENTRECONNECTTIMEOUT</u>	Y	Y	Y	N/D	N/D
<u>CLIENTID</u>	N/D	Y	N/D	N/D	N/D
<u>COMPHDR</u> "1" na página 54	Y	N/D	Y	N/D	N/D
<u>COMPMSG</u> "1" na página 54	Y	Y	Y	N/D	N/D
<u>CONNOPT</u> "1" na página 54	Y	Y	Y	N/D	N/D
<u>CONNTAG</u> "1" na página 54	Y	Y	Y	N/D	N/D
<u>DESCRIPTION</u> "1" na página 54	N/D	Y	N/D	Y	Y
<u>Expiração</u> "1" na página 54	N/D	N/D	N/D	Y	Y
<u>FAILIFQUIESCE</u>	Y	Y	Y	Y	Y
<u>HOSTNAME</u>	N/D	Y	N/D	N/D	N/D

Tabela 9. Propriedades do XMS .NET (continuação)

Propriedade	Tipo de Objeto				
	CF	QCF	TCF	Fila	Tópico
<u>LOCALADDRESS</u>	N/D	Y	N/D	N/D	N/D
<u>Persistência</u>	N/D	N/D	N/D	Y	Y
<u>SOAP</u>	N/D	Y	N/D	N/D	N/D
<u>Prioridade "1" na página 54</u>	N/D	N/D	N/D	Y	Y
<u>PROVIDERVERSION "1" na página 54</u>	N/D	Y	N/D	N/D	N/D
<u>QMANAGER</u>	Y	Y	Y	Y	N/D
<u>QUEUE "1" na página 54</u>	N/D	N/D	N/D	Y	N/D
<u>SHARECONVALLOWED</u>	Y	Y	Y	N/D	N/D
<u>tópico "1" na página 54</u>	N/D	N/D	N/D	N/D	Y
<u>Transporte "1" na página 54</u>	N/D	Y	N/D	N/D	N/D

**Nota:**

1. Essas propriedades não possuem propriedades de nível de aplicativo mas opcionalmente podem ser definidas usando propriedades administradas.

**Conceitos relacionados**

Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

## Tarefas relacionadas

### Criando Objetos Administrados

As definições de objeto `ConnectionFactory` e `Destination` que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### Objetos `InitialContext`

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

## Referências relacionadas

### Propriedades necessárias para objetos `ConnectionFactory` administrados

Quando um aplicativo cria um `connection factory`, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto `Destination` administrado.

### `IDestination` (para a interface `.NET`)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

### Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

### `IConnectionFactory` (para a interface `.NET`)

Um aplicativo usa um `connection factory` para criar uma conexão.

### Propriedades de `ConnectionFactory`

Uma visão geral das propriedades do objeto `ConnectionFactory`, com links para informações de referência mais detalhadas.

## Propriedades necessárias para objetos `ConnectionFactory` administrados

Quando um aplicativo cria um `connection factory`, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

As propriedades listadas nas tabelas a seguir são o mínimo necessário para que um aplicativo seja configurado para criar uma conexão com um servidor de sistema de mensagens. Se você desejar customizar a maneira que uma conexão é criada, seu aplicativo poderá configurar quaisquer propriedades adicionais do objeto `ConnectionFactory`, conforme necessário. Para obter mais informações, consulte [“Propriedades de `ConnectionFactory`” na página 182](#). Uma lista completa de propriedades disponíveis está incluída.

## Conexão com um gerenciador de filas do IBM MQ

<b>Necessário XMS</b>	<b>Propriedade equivalente do IBM MQ JMS necessária</b>
<u><code>XMSC_CONNECTION_TYPE</code></u>	XMS trabalha isso a partir do nome da classe do <code>connection factory</code> e da propriedade <code>TRANSPORT (TRAN)</code> .
<u><code>XMSC_WMQ_HOST_NAME</code></u>	<code>HOSTNAME (HOST)</code>
<u><code>XMSC_WMQ_PORT</code></u>	<code>PORTA</code>
<u><code>XMSC_WMQ_QUEUE_MANAGER</code></u>	Nome do Gerenciador de Filas

## Conexão de tempo real com um broker

<i>Tabela 11. Configurações de propriedade para objetos ConnectionFactory administrados para conexões em tempo real com um broker</i>	
<b>Necessário XMS</b>	<b>Propriedade equivalente do IBM MQ JMS necessária</b>
<u>XMSC_CONNECTION_TYPE</u>	XMS trabalha isso a partir do nome da classe do connection factory e da propriedade TRANSPORT (TRAN).
<u>XMSC_RTT_HOST_NAME</u>	HOSTNAME (HOST)
<u>XMSC_RTT_PORT</u>	PORTA

## Conexão com um WebSphere Application Server service integration bus

<i>Tabela 12. Configurações de propriedade para objetos ConnectionFactory administrados para conexões com um WebSphere Application Server service integration bus</i>	
<b>Propriedade XMS</b>	<b>Descrição</b>
<u>XMSC_CONNECTION_TYPE</u>	O tipo de servidor de mensagens para o qual um aplicativo se conecta.. Isso é determinado a partir do nome da classe do connection factory.
<u>XMSC_WPM_BUS_NAME</u>	Para um connection factory, o nome do barramento de integração de serviços ao qual o aplicativo se conecta ou, para um destino, o nome do barramento de integração de serviços no qual o destino existe.

### Conceitos relacionados

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

#### Conexões seguras com um gerenciador de filas IBM MQ

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um gerenciador de filas IBM MQ , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### Conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um mecanismo do sistema de mensagens WebSphere Application Server service integration bus , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### **Tarefas relacionadas**

##### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

##### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

#### **Referências relacionadas**

##### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

##### ConnectionFactory (para a interface .NET)

Um aplicativo usa um connection factory para criar uma conexão.

##### Propriedades de ConnectionFactory

Uma visão geral das propriedades do objeto ConnectionFactory , com links para informações de referência mais detalhadas.

## **Propriedades Necessárias para Objetos de Destino Administrados**

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

<b>Tipo de conexão</b>	<b>Propriedade</b>	<b>Descrição</b>
Gerenciador de filas da IBM MQ	FILA (QU) TÓPICO (SUPERIOR)	A fila à qual você deseja se conectar O tópico que o aplicativo usa como um destino
Conexão de tempo real com um broker	TÓPICO (SUPERIOR)	O tópico que o aplicativo usa como um destino
WebSphere Application Server service integration bus	topicName queueName	Se seu aplicativo estiver se conectando a um tópico Se seu aplicativo estiver se conectando a uma fila

#### **Conceitos relacionados**

##### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

##### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

## **Tarefas relacionadas**

### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

## **Referências relacionadas**

### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

### IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

### Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

## **Criando Objetos Administrados**

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

## **Antes de começar**

Para obter detalhes adicionais sobre os diferentes tipos de repositório de objeto administrado que o XMS suporta, consulte [“Tipos Suportados de Repositório de Objetos Administrados” na página 51](#).

## **Sobre esta tarefa**

Para criar os objetos administrados do IBM MQ, use a ferramenta de administração JMS (JMSAdmin) do IBM MQ Explorer ou IBM MQ.

Para criar os objetos administrados para IBM MQ ou IBM Integration Bus, use a ferramenta IBM MQ de administração JMS (JMSAdmin).

Para criar objetos administrados para WebSphere Application Server service integration bus, use o console administrativo WebSphere Application Server .

As etapas a seguir resumem o que você faz para criar objetos administrados.

## Procedimento

1. Crie um connection factory e defina as propriedades necessárias para criar uma conexão de seu aplicativo com o servidor escolhido.

As propriedades mínimas que o XMS requer para fazer uma conexão são definidas em [“Propriedades necessárias para objetos ConnectionFactory administrados”](#) na página 55.

2. Crie o destino necessário no servidor de sistema de mensagens, ao qual seu aplicativo se conecta:

- Para uma conexão com um gerenciador de filas do IBM MQ, crie uma fila ou um tópico.
- Para uma conexão em tempo real com um broker, crie um tópico.
- Para uma conexão com um WebSphere Application Server service integration bus, crie uma fila ou um tópico.

As propriedades mínimas que o XMS requer para fazer uma conexão são definidas em [“Propriedades Necessárias para Objetos de Destino Administrados”](#) na página 57.

## Conceitos relacionados

### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

### Objetos Administrados

Usando objetos administrados, é possível administrar as configurações de conexão usadas pelos aplicativos clientes a serem administradas a partir de um repositório central. Um aplicativo recupera definições de objeto do repositório central e as utiliza para criar objetos ConnectionFactory e Destination . Usando objetos administrados, é possível desacoplar aplicativos dos recursos usados no tempo de execução.

### ConnectionFactories e objetos de Conexão

Um objeto ConnectionFactory fornece um modelo que um aplicativo usa para criar um objeto Connection. O aplicativo usa o objeto Connection para criar um objeto Session.

### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

### **Tarefas relacionadas**

#### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

#### IConnectionFactory (para a interface .NET)

Um aplicativo usa um connection factory para criar uma conexão.

#### Propriedades de ConnectionFactory

Uma visão geral das propriedades do objeto ConnectionFactory , com links para informações de referência mais detalhadas.

#### IDestination (para a interface .NET)

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

#### Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

## **Objetos InitialContext**

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

### **Sobre esta tarefa**

Um objeto InitialContext encapsula uma conexão com o repositório. A API XMS fornece métodos para executar as tarefas a seguir:

- Criar um objeto InitialContext
- Consultar um objeto administrado no repositório de objetos administrado.

Para obter detalhes adicionais sobre a criação de um objeto InitialContext , consulte [“InitialContext” na página 113](#) para .REDE e [“Propriedades de InitialContext” na página 192](#)

### **Conceitos relacionados**

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão

com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

#### InitialContext (para a interface .NET)

Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.

#### Propriedades de InitialContext

Uma visão geral das propriedades do objeto InitialContext com links para informações de referência mais detalhadas.

## **Propriedades InitialContext**

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

No JNDI e na implementação .NET de XMS, as informações adicionais são fornecidas em um ambiente Hashtable ao construtor.

O local do repositório de objetos administrado é definido na propriedade XMSC\_IC\_URL . Normalmente, essa propriedade é transmitida na chamada Criar, mas pode ser modificada para se conectar a um diretório de nomenclatura diferente antes da consulta. Para contextos FileSystem ou LDAP, essa propriedade define o endereço do diretório. Para nomenclatura COS, esse é o endereço do serviço da web que usa essas propriedades para se conectar ao diretório JNDI.

As propriedades a seguir são transmitidas sem modificação para o serviço da web que as usará para usar para se conectar ao diretório JNDI.

- XMSC\_IC\_PROVIDER\_URL
- XMSC\_IC\_SECURITY\_CREDENTIALS
- XMSC\_IC\_SECURITY\_AUTHENTICATION
- XMSC\_IC\_SECURITY\_PRINCIPAL

- [XMSC\\_IC\\_SECURITY\\_PROTOCOL](#)

### **Conceitos relacionados**

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

#### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

#### InitialContext (para a interface .NET)

Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.

#### Propriedades de InitialContext

Uma visão geral das propriedades do objeto InitialContext com links para informações de referência mais detalhadas.

## **Formato de URI para contextos iniciais XMS**

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

### **Contexto de FileSystem**

Para o contexto FileSystem, a URL fornece o local do diretório baseado no sistema de arquivos. A estrutura da URL é conforme definido pelo RFC 1738, *Localizadores Uniformes de Recursos (URL)*: a URL

tem o prefixo `file://` e a sintaxe após esse prefixo é uma definição válida de um arquivo que pode ser aberto no sistema no qual o XMS está executando.

Esta sintaxe pode ser específica da plataforma e pode usar os separadores '/' separadores ou '\'. Se você usar '\', cada separador precisará ser escapado usando um '\\' adicional. Isso impede que a estrutura .NET tente interpretar o separador como um caractere de escape para o que segue.

Esses exemplos ilustram esta sintaxe:

```
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 o contexto LDAP, a estrutura básica da URL é conforme definido pelo RFC 2255, *The LDAP URL Format*, com o prefixo sem distinção entre maiúsculas e minúsculas `ldap://`

A sintaxe exata é ilustrada no exemplo a seguir:

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

Essa sintaxe é definida no RFC, mas sem suporte para quaisquer atributos, escopo, filtros ou extensões.

Exemplos desta sintaxe incluem:

```
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 de WSS

Para o contexto WSS, a URL está no formato de um terminal de serviços da Web, com o prefixo `http://`.

Como alternativa, é possível usar o prefixo `cosnaming://` ou `wsvc://`,

Essas duas prefixos são interpretadas como significando que você está usando um contexto WSS com a URL acessada através de http, o que permite que o tipo de contexto inicial seja derivado facilmente diretamente da URL.

Exemplos dessa sintaxe incluem o seguinte:

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

## Conceitos relacionados

### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

#### **Tarefas relacionadas**

##### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

##### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

#### **Referências relacionadas**

##### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

##### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

##### InitialContext (para a interface .NET)

Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.

##### Propriedades de InitialContext

Uma visão geral das propriedades do objeto InitialContext com links para informações de referência mais detalhadas.

## **Serviço da Web de Consulta de JNDI**

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

O serviço da Web é fornecido no arquivo de archive corporativo SIBXJndiLookupEAR.ear, localizado dentro do diretório de instalação. Para a liberação atual de IBM Message Service Client for .NET, SIBXJndiLookupEAR.ear pode ser localizado no diretório *install\_dir*\java\lib . Isso pode ser instalado em um servidor WebSphere Application Server service integration bus , usando o console administrativo ou a ferramenta de script wsadmin. Consulte a documentação do produto para obter informações adicionais sobre como implementar aplicativos de serviço da web.

Para definir o serviço da web dentro de aplicativos XMS , você simplesmente precisa configurar a propriedade XMSC\_IC\_URL do objeto InitialContext para a URL do terminal de serviço da web. Por exemplo, se o serviço da web for implementado em um host do servidor chamado MyServer, um exemplo de uma URL de terminal de serviço da web:

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

Configurar a propriedade XMSC\_IC\_URL permite que chamadas InitialContext Lookup chamem o serviço da web no terminal definido que, por sua vez, consulta o objeto administrado necessário por meio do serviço de nomenclatura COS.

Os aplicativos .NET podem usar o serviço da web. A implementação do lado do servidor é a mesma para XMS C, /C++ e, XMS.NET.XMS .NET chama o serviço da web diretamente por meio da estrutura Microsoft .NET .

### **Conceitos relacionados**

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

#### Configurando o Ambiente do Servidor de Mensagens

Os tópicos nesta seção descrevem como configurar o ambiente do servidor de sistema de mensagens para permitir que os aplicativos XMS se conectem a um servidor.

#### Usando os aplicativos de amostra XMS

Use os aplicativos de amostra fornecidos com o XMS para verificar sua instalação e configuração do servidor de sistema de mensagens e para ajudá-lo a construir seus aplicativos. As amostras fornecem uma visão geral dos recursos comuns de cada API.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

#### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

#### Instalando Message Service Client for .NET usando o assistente de instalação

A instalação usa um instalador MSI InstallShield X/Windows. Duas opções de configuração estão disponíveis, para que seja possível escolher uma instalação completa ou customizada.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

## Recuperação de Objetos Administrados

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

Os objetos a serem recuperados podem ter os seguintes tipos de nomes:

- Um nome simples que descreve o objeto Destination, por exemplo, um destino de fila chamado SalesOrders
- Um nome composto, que pode ser composto de SubContextos, separados por '/' e deve terminar com o nome do objeto. Um exemplo de um nome composto é "Warehouse / PickLists/DispatchQueue2", em que Warehouse e Picklists são SubContextos no diretório de nomenclatura e DispatchQueue2 é o nome de um objeto de Destino.

### Conceitos relacionados

#### Tipos Suportados de Repositório de Objetos Administrados

Os objetos administrados do Sistema de Arquivos e do LDAP podem ser usados para se conectar ao IBM MQ e ao WebSphere Application Server, enquanto que o COS Naming pode ser usado para se conectar apenas ao WebSphere Application Server.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

#### Propriedades InitialContext

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### Formato de URI para contextos iniciais XMS

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### Serviço da Web de Consulta de JNDI

Para acessar um diretório de nomenclatura COS a partir de XMS, um serviço da web JNDI Lookup deve ser implementado em um servidor WebSphere Application Server service integration bus . Esse serviço da web converte as informações de Java do serviço de nomenclatura COS em um formulário que os aplicativos XMS podem ler.

### Tarefas relacionadas

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

#### Objetos InitialContext

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

### Referências relacionadas

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

### InitialContext (para a interface .NET)

Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.

### Propriedades de InitialContext

Uma visão geral das propriedades do objeto InitialContext com links para informações de referência mais detalhadas.

## **Protegendo Comunicações para Aplicativos XMS**

Esta seção fornece informações sobre a configuração de comunicações seguras para permitir que os aplicativos XMS se conectem por meio de Secure Sockets Layer (SSL) para um mecanismo de sistema de mensagens WebSphere Application Server service integration bus ou gerenciador de filas IBM MQ .

A seção contém os seguintes tópicos:

- [“Conexões seguras com um gerenciador de filas IBM MQ” na página 67](#)
- [“Mapeamentos de nomes CipherSuite e CipherSpec para conexões com um gerenciador de filas do IBM MQ” na página 68](#)
- [“Conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus” na página 69](#)
- [“Mapeamentos de Nome CipherSuite e CipherSpec para Conexões com um WebSphere Application Server service integration bus” na página 70](#)

### **Conexões seguras com um gerenciador de filas IBM MQ**

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um gerenciador de filas IBM MQ , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

O protocolo usado na negociação de criptografia pode ser Secure Sockets Layer (SSL) ou Segurança da Camada de Transporte (TLS), dependendo de qual CipherSuite você especificar no objeto ConnectionFactory.

Se você usar as bibliotecas do cliente do IBM WebSphere MQ 7.0.0 Fix Pack 1 e mais recentes e conectar-se a um gerenciador de filas do IBM WebSphere MQ 7.0, será possível criar várias conexões com o mesmo gerenciador de filas no aplicativo XMS. No entanto, a conexão com um gerenciador de filas diferente não é permitida. Se você tentar, receberá o erro MQRC\_SSL\_ALREADY\_INITIALIZED .

Se você usar as bibliotecas do cliente IBM WebSphere MQ 6.0 e mais recente, será possível criar uma conexão SSL apenas se você fechar qualquer conexão SSL anterior primeiro. Várias conexões SSL simultâneas a partir do mesmo processo para os mesmos gerenciadores de filas ou gerenciadores de filas diferentes não são permitidas. Se você tentar mais de uma solicitação, receberá o aviso MQRC\_SSL\_ALREADY\_INITIALIZED, o que pode significar que alguns parâmetros solicitados para a conexão SSL foram ignorados.

As propriedades do ConnectionFactory para conexões por meio de SSL com um gerenciador de filas do IBM MQ, com uma breve descrição, são mostradas na tabela a seguir:

<b>Nome da propriedade</b>	<b>Descrição</b>
<u>XMSC_WMQ_SSL_CERT_STORES</u>	Os locais dos servidores que retêm as listas de revogação de certificado (CRLs) a serem usadas em uma conexão SSL com um gerenciador de filas.
<u>XMSC_WMQ_SSL_CIPHER_SPEC</u>	O nome da CipherSpec a ser usada em uma conexão segura com um gerenciador de filas.

Tabela 14. Propriedades de ConnectionFactory para conexões com um gerenciador de filas IBM MQ via SSL (continuação)

Nome da propriedade	Descrição
<u>XMSC_WMQ_SSL_CIPHER_SUITE</u>	O nome do CipherSuite a ser usado em uma conexão TLS para um gerenciador de filas. O protocolo usado para negociar a conexão segura depende do CipherSuite especificado.
<u>XMSC_WMQ_SSL_CRYPT_HW</u>	Detalhes de configuração para o hardware criptográfico conectado ao sistema do cliente.
<u>XMSC_WMQ_SSL_FIPS_REQUIRED</u>	O valor dessa propriedade determina se um aplicativo pode ou não usar conjuntos de cifras compatíveis não FIPS. Se essa propriedade for configurada como true, apenas algoritmos do FIPS serão usados para a conexão cliente-servidor.
<u>XMSC_WMQ_SSL_KEY_REPOSITORY</u>	O local do arquivo do banco de dados de chaves no qual chaves e certificados são armazenados.
<u>XMSC_WMQ_SSL_KEY_RESETCOUNT</u>	O KeyResetCount representa o número total de bytes não criptografados enviados e recebidos dentro de uma conversa SSL antes de a chave secreta ser renegociada.
<u>XMSC_WMQ_SSL_PEER_NAME</u>	O nome do peer a ser usado em uma conexão SSL com um gerenciador de filas.

#### Referências relacionadas

IConnectionFactory (para a interface .NET)

Um aplicativo usa um connection factory para criar uma conexão.

Propriedades de ConnectionFactory

Uma visão geral das propriedades do objeto ConnectionFactory, com links para informações de referência mais detalhadas.

Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

#### **Mapeamentos de nomes CipherSuite e CipherSpec para conexões com um gerenciador de filas do IBM MQ**

O InitialContext é convertido entre a propriedade SSLCIPHERSUITE da Connection Factory de JMSAdmin e o XMS próximo-equivalente XMSC\_WMQ\_SSL\_CIPHER\_SPEC. Uma conversão semelhante é necessária se você especificar um valor para XMSC\_WMQ\_SSL\_CIPHER\_SUITE mas omitir valor para XMSC\_WMQ\_SSL\_CIPHER\_SPEC.

Tabela 15 na página 68 lista os CipherSpecs disponíveis e seus equivalentes JSSE CipherSuite.

Tabela 15. CipherSpecs Disponíveis e seus Equivalentes JSSE CipherSuite

CipherSpec	Equivalente CipherSuite JSSE
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

**Nota:**

- TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA foi descontinuado. No entanto, ele ainda pode ser usado para transferir até 32 GB de dados antes de a conexão ser finalizada com erro AMQ9288. Para evitar esse erro, você precisará evitar o uso de DES triplo ou ativar a reconfiguração de chave secreta ao usar esse CipherSpec.

## Conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um mecanismo do sistema de mensagens WebSphere Application Server service integration bus , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

XMS fornece suporte SSL e HTTPS para conexões com um WebSphere Serviço Integração Bus. SSL e HTTPS fornecem conexões seguras para autenticação e confidencialidade.

Assim como a segurança WebSphere , a segurança XMS é configurada com relação aos padrões de segurança JSSE e convenções de nomenclatura, que incluem o uso de CipherSuites para especificar os algoritmos que são usados ao negociar uma conexão segura. O protocolo usado na negociação de criptografia pode ser SSL ou TLS, dependendo de qual CipherSuite você especifica no objeto ConnectionFactory.

Tabela 16 na página 69 lista as propriedades que devem ser definidas no objeto ConnectionFactory.

<i>Tabela 16. Propriedades de ConnectionFactory para Conexões Seguras com um Mecanismo de Sistema de Mensagens WebSphere Application Server service integration bus</i>	
<b>Nome da propriedade</b>	<b>Descrição</b>
<u>XMSC_WPM_SSL_CIPHER_SUITE</u>	O nome do CipherSuite a ser usado em uma conexão TLS com um mecanismo do sistema de mensagens do WebSphere Serviço Integração Bus O protocolo usado para negociar a conexão segura depende do CipherSuite especificado.
<u>XMSC_WPM_SSL_KEYRING_LABEL</u>	O certificado a ser usado ao autenticar-se com o servidor.

A seguir está um exemplo de propriedades ConnectionFactory para conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus:

```
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);
```

Em que chain\_name deve ser configurado como BootstrapTunneledeledSecureMessaging ou BootstrapSecureMessaging e port\_number é o número da porta na qual o servidor de autoinicialização atende a solicitações recebidas.

A seguir, está um exemplo de propriedades ConnectionFactory para conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus com valores de amostra inseridos:

```
/* 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");
```

### Referências relacionadas

[IConnectionFactory](#) (para a interface .NET)

Um aplicativo usa um connection factory para criar uma conexão.

#### Propriedades de ConnectionFactory

Uma visão geral das propriedades do objeto ConnectionFactory , com links para informações de referência mais detalhadas.

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

### **Mapeamentos de Nome CipherSuite e CipherSpec para Conexões com um WebSphere Application Server service integration bus**

Como o GSKit usa CipherSpecs em vez de CipherSuites, os nomes de CipherSuite de estilo JSSE especificados na propriedade XMSC\_WPM\_SSL\_CIPHER\_SUITE devem ser mapeados para os nomes de CipherSpec de estilo GSKit.

Tabela 17 na página 70 lista o CipherSpec equivalente para cada CipherSuite reconhecido.

<b>CipherSuite</b>	<b>CipherSpec equivalente</b>
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 foi descontinuado. No entanto, ele ainda pode ser usado para transferir até 32 GB de dados antes de a conexão ser finalizada com erro AMQ9288. Para evitar esse erro, você precisará evitar o uso de DES triplo ou ativar a reconfiguração de chave secreta ao usar esse CipherSpec.

## **Mensagens do XMS**

Esta seção descreve a estrutura e o conteúdo de mensagens XMS e explica como os aplicativos processam mensagens XMS .

Esta seção contém os seguintes tópicos:

- [“Partes de uma mensagem XMS” na página 70](#)
- [“Campos de cabeçalho em uma mensagem XMS” na página 71](#)
- [“Propriedades de uma mensagem XMS” na página 72](#)
- [“O corpo de uma mensagem XMS” na página 76](#)
- [“Seletores de mensagens” na página 82](#)
- [“Mapeando mensagens do XMS para mensagens do IBM MQ” na página 83](#)

#### **Referências relacionadas**

[IMessage](#) (para a interface .NET)

Um objeto de mensagem representa uma mensagem que um aplicativo envia ou recebe. IMessage é uma superclasse para as classes de mensagem, como IMapMessage.

## **Partes de uma mensagem XMS**

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

## **Cabeçalho**

O cabeçalho de uma mensagem contém campos, e todas as mensagens contêm o mesmo conjunto de campos de cabeçalho. XMS e aplicativos usam os valores dos campos de cabeçalho para identificar e rotear mensagens. Para obter mais informações sobre campos de cabeçalho, consulte [“Campos de cabeçalho em uma mensagem XMS”](#) na página 71.

## **Conjunto de propriedades**

As propriedades de uma mensagem especificam informações adicionais sobre a mensagem. Embora todas as mensagens tenham o mesmo conjunto de campos de cabeçalho, cada mensagem pode ter um conjunto diferente de propriedades. Para obter mais informações, consulte [“Propriedades de uma mensagem XMS”](#) na página 72.

## **Conteúdo**

O corpo de uma mensagem contém dados do aplicativo. Para obter mais informações, consulte [“O corpo de uma mensagem XMS”](#) na página 76.

Um aplicativo pode selecionar quais mensagens ele deseja receber. Usando seletores de mensagens, que especificam os critérios de seleção. Os critérios podem ser baseados nos valores de determinados campos de cabeçalho e os valores de qualquer uma das propriedades de uma mensagem. Para obter mais informações sobre os seletores de mensagem, veja [“Seletores de mensagens”](#) na página 82.

## **Referências relacionadas**

### Campos de cabeçalho em uma mensagem XMS

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS , o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS .

### Propriedades de uma mensagem XMS

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

### Seletores de mensagens

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

### Mapeando mensagens do XMS para mensagens do IBM MQ

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

## **Campos de cabeçalho em uma mensagem XMS**

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS , o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS .

Os nomes desses campos de cabeçalho começam com o prefixo JMS.. Para obter uma descrição dos campos do cabeçalho da mensagem JMS, consulte *Java Message Service Especificação*.

XMS implementa os campos de cabeçalho da mensagem JMS como atributos de um objeto de Mensagem. Cada campo de cabeçalho possui seus próprios métodos para configurar e obter seu valor. Para uma descrição desses métodos, consulte [“IMessage”](#) na página 125. Um campo de cabeçalho é sempre legível e gravável.

[Tabela 18 na página 72](#) lista os campos de cabeçalho da mensagem JMS e indica como o valor de cada campo é configurado para uma mensagem transmitida. Alguns dos campos são configurados automaticamente por XMS quando um aplicativo envia uma mensagem ou, no caso de JMSRedelivered, quando um aplicativo recebe uma mensagem.

Tabela 18. Campos de cabeçalho da mensagem JMS

Nome do campo de cabeçalho de mensagem do JMS	Como o valor é configurado para uma mensagem transmitida (no formato <i>method [class]</i> )
JMSCorrelationID	Set JMSCorrelationID [ Message ]
JMSDeliveryMode	Send [ MessageProducer ]
JMSDestination	Send [ MessageProducer ]
JMSExpiration	Send [ MessageProducer ]
JMSMessageID	Send [ MessageProducer ]
JMSPriority	Send [ MessageProducer ]
JMSRedelivered	Receive [ MessageConsumer ]
JMSReplyTo	Set JMSReplyTo [ Message ]
JMSTimestamp	Send [ MessageProducer ]
JMSType	Set JMSType [ Message ]

### Referências relacionadas

#### Partes de uma mensagem XMS

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

#### Propriedades de uma mensagem XMS

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

#### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

#### Seletores de mensagens

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

#### Mapeando mensagens do XMS para mensagens do IBM MQ

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

## Propriedades de uma mensagem XMS

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

Um aplicativo XMS pode trocar mensagens com um aplicativo WebSphereJMS porque o XMS suporta as propriedades predefinidas a seguir de um objeto de Mensagem:

- As mesmas propriedades definidas pelo JMS que o WebSphereJMS suporta. Os nomes dessas propriedades começam com o prefixo JMSX.
- As mesmas propriedades definidas pelo IBM que o WebSphereJMS suporta. Os nomes dessas propriedades começam com o prefixo JMS\_IBM\_.

Cada propriedade predefinida possui dois nomes:

- Um nome JMS , para uma propriedade JMSdefinida ou um nome WebSphereJMS , para uma propriedade IBMdefinida.

Este é o nome pelo qual a propriedade é conhecida em JMS ou WebSphere JMS, e é também o nome que é transmitido com uma mensagem que tem esta propriedade. Um aplicativo XMS usa este nome para identificar a propriedade em uma expressão de seletor de mensagem.

- Um nome de XMS para identificar a propriedade em todas as situações, exceto em uma expressão de seletor de mensagem. Cada nome XMS é definido como uma constante nomeada na classe `IBM.XMS.XMSC`. O valor da constante nomeada é o nome JMS ou WebSphereJMS correspondente.

Além das propriedades predefinidas, um aplicativo XMS pode criar e usar seu próprio conjunto de propriedades de mensagem. Essas propriedades são chamadas de *propriedades definidas pelo aplicativo*.

Depois que um aplicativo cria uma mensagem, as propriedades da mensagem são legíveis e graváveis. As propriedades permanecem legíveis e graváveis depois que o aplicativo envia a mensagem. Quando um aplicativo recebe uma mensagem, as propriedades da mensagem são somente leitura. Se um aplicativo chamar o método `Clear Properties` da classe de Mensagem quando as propriedades de uma mensagem forem somente leitura, as propriedades se tornarão legíveis e graváveis. O método também limpa as propriedades.

A mensagem recebida, quando encaminhada após a limpeza das propriedades da mensagem, se comportará de uma maneira consistente com o comportamento de encaminhamento de um `BytesMessage XMS for .NET` do WMQ padrão com propriedades de mensagem limpas.

Isso, no entanto, não é recomendado desde que as seguintes propriedades sejam perdidas:

- Valor da propriedade `JMS_IBM_Encoding`, implicando que os dados da mensagem não podem ser decodificados de forma significativa.
- Valor da propriedade `JMS_IBM_Format`, implicando que o encadeamento de cabeçalho entre o cabeçalho da mensagem (MQMD ou o novo MQRFH2) e os cabeçalhos existentes seriam quebrados.

Para determinar os valores de todas as propriedades de uma mensagem, um aplicativo pode chamar o método `Get Properties` da classe `Message`. O método cria um agente iterativo que encapsula uma lista de objetos de Propriedade, em que cada objeto Propriedade representa uma propriedade da mensagem. O aplicativo pode então usar os métodos da classe `Iterator` para recuperar cada objeto de Propriedade por vez, e ele pode usar os métodos da classe de Propriedade para recuperar o nome, o tipo de dados e o valor de cada propriedade.

## Referências relacionadas

### Partes de uma mensagem XMS

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

### Campos de cabeçalho em uma mensagem XMS

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS, o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS.

### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

### Seletores de mensagens

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

### Mapeando mensagens do XMS para mensagens do IBM MQ

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

## **JMS-propriedades definidas de uma mensagem**

Várias propriedades definidas pelo JMS de uma mensagem são suportadas por XMS e WebSphereJMS.

Tabela 19 na página 74 lista as propriedades definidas por JMS de uma mensagem que são suportadas por ambos XMS e WebSphere JMS. Para obter uma descrição das propriedades JMS-definidas, consulte *Especificação de Java Message Service*. As propriedades JMS-definidas não são válidas para uma conexão em tempo real com um broker.

A tabela especifica o tipo de dados de cada propriedade e indica como o valor da propriedade é configurado para uma mensagem transmitida. Algumas das propriedades são configuradas automaticamente por XMS quando um aplicativo envia uma mensagem ou, no caso de `JMSXDeliveryCount`, quando um aplicativo recebe uma mensagem.

Tabela 19. JMS-propriedades definidas de uma mensagem

Nome XMS da propriedade JMS definida	JMS nome	Tipo de Dados	Como o valor é configurado para uma mensagem transmitida (no formato <i>method [class]</i> )
JMSX_APPID	JMSXAppID	System.String	Send [ MessageProducer ]
JMSX_DELIVERY_COUNT	JMSXDeliveryCount	System.Int32	Receive [ MessageConsumer ]
JMSX_GROUPID	JMSXGroupID	System.String	Configurar propriedade de sequência [PropertyContext]
JMSX_GROUPSEQ	JMSXGroupSeq	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMSX_USERID	JMSXUserID	System.String	Send [ MessageProducer ]

### IBM-propriedades definidas de uma mensagem

Várias propriedades definidas pela IBM de uma mensagem são suportadas por XMS e WebSphereJMS.

Tabela 20 na página 74 lista as propriedades definidas do IBM de uma mensagem que são suportadas por XMS e WebSphereJMS. Para obter mais informações sobre as propriedades IBM definidas, consulte a documentação do produto IBM MQ ou WebSphere Application Server .

A tabela especifica o tipo de dados de cada propriedade e indica como o valor da propriedade é configurado para uma mensagem transmitida. Algumas das propriedades são configuradas automaticamente por XMS quando um aplicativo envia uma mensagem.

Tabela 20. Propriedades Definidas pela IBM de uma Mensagem

Nome XMS da propriedade IBM definida	WebSphere JMS nome	Tipo de Dados	Como o valor é configurado para uma mensagem transmitida (no formato <i>method [class]</i> )
JMS_IBM_CHARACTER_SET	JMS_IBM_Character_Set	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
CODIFICAÇÃO DE JMS_IBM_ENCODING	JMS_IBM_Encoding	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_EXCEPTIONMESSAGE	JMS_IBM_ExceptionMessage	System.String	Receive [ MessageConsumer ]
JMS_IBM_EXCEPTIONREASON	JMS_IBM_ExceptionReason	System.Int32	Receive [ MessageConsumer ]
JMS_IBM_EXCEPTIONTIMESTAMP	JMS_IBM_ExceptionTimestamp	System.Int64	Receive [ MessageConsumer ]
JMS_IBM_EXCEPTIONPROBLEMDESTINO:	JMS_IBM_ExceptionProblemDestination	System.String	Receive [ MessageConsumer ]

Tabela 20. Propriedades Definidas pela IBM de uma Mensagem (continuação)

<b>Nome XMS da propriedade IBM definida</b>	<b>WebSphere JMS nome</b>	<b>Tipo de Dados</b>	<b>Como o valor é configurado para uma mensagem transmitida (no formato <i>method [class]</i>)</b>
JMS_IBM_FEEDBACK	JMS_IBM_Feedback	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
FORMATO JMS_IBM_FORMAT	JMS_IBM_Format	System.String	Configurar propriedade de sequência [PropertyContext]
JMS_IBM_LAST_MSG_IN_GROUP	JMS_IBM_Last_Msg_In_Group	System.Boolean	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_MSGTYPE	JMS_IBM_MsgType	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_PUTAPPLTYPE	JMS_IBM_PutApplType	System.Int32	Send [ MessageProducer ]
JMS_IBM_PUTDATE	JMS_IBM_PutDate	System.String	Send [ MessageProducer ]
JMS_IBM_PUTTIME	JMS_IBM_PutTime	System.String	Send [ MessageProducer ]
JMS_IBM_REPORT_COA	JMS_IBM_Report_COA	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_COD	JMS_IBM_Report_COD	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_DISCARD_MSG	JMS_IBM_Report_Discard_Msg	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_EXCEPTION	JMS_IBM_Report_Exception	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_EXPIRATION	JMS_IBM_Report_Expiration	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_NAN	JMS_IBM_Report_NAN	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_PAN	JMS_IBM_Report_PAN	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_REPORT_PASS_CORREL_ID	JMS_IBM_Report_Pass_Correl_ID	System.Int32	Configurar Propriedade Integer [ PropertyContext ]

Tabela 20. Propriedades Definidas pela IBM de uma Mensagem (continuação)

Nome XMS da propriedade IBM definida	WebSphere JMS nome	Tipo de Dados	Como o valor é configurado para uma mensagem transmitida (no formato <i>method [class]</i> )
JMS_IBM_REPORT_PASS_MSG_ID	JMS_IBM_Report_Pass_Msg_ID	System.Int32	Configurar Propriedade Integer [ PropertyContext ]
JMS_IBM_SYSTEM_MESS AGEID	JMS_IBM_System_Messa geID	System.String	Send [ MessageProducer ]

### Propriedades Definidas pelo Aplicativo de uma Mensagem

Um aplicativo XMS pode criar e usar seu próprio conjunto de propriedades de mensagem. Quando um aplicativo envia uma mensagem, essas propriedades também são transmitidas com a mensagem. Um aplicativo de recebimento, usando seletores de mensagens, pode, então, selecionar quais mensagens ele deseja receber com base nos valores dessas propriedades.

Para permitir que um aplicativo WebSphere JMS selecione e processe mensagens enviadas por um aplicativo XMS, o nome de uma propriedade definida pelo aplicativo deve estar em conformidade com as regras para formar identificadores em expressões do seletor de mensagens, conforme documentado na documentação do produto IBM MQ. O valor de uma propriedade de aplicativo definido deve ter um dos seguintes tipos de dados: System.Boolean, System.SByte, System.Int16, System.Int32, System.Int64, System.Float, System.Double ou System.String.

### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

XMS suporta cinco tipos de corpo da mensagem:

#### bytes

O corpo contém um fluxo de bytes. Uma mensagem com esse tipo de corpo é chamada de mensagem de *bytes*. A interface `IBytesMessage` contém os métodos para processar o corpo de uma mensagem de bytes. Para mais informações, consulte [“Mensagens de Bytes” na página 78](#).

#### Mapear

O corpo contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado. Uma mensagem com esse tipo de corpo é chamada de *mensagem de mapa*. A interface `IMapMessage` contém os métodos para processar o corpo de uma mensagem de mapa. Para mais informações, consulte [“Mensagens de Mapa” na página 79](#).

#### Object

O corpo contém um objeto serializado Java ou .NET. Uma mensagem com esse tipo de corpo é chamada de mensagem de objeto. A interface `IObjectMessage` contém os métodos para processar o corpo de uma mensagem de objeto. Para mais informações, consulte [“Mensagens de objeto” na página 80](#).

#### Fluxo

O corpo contém um fluxo de valores, em que cada valor possui um tipo de dados associado. Uma mensagem com esse tipo de corpo é chamada de *mensagem de fluxo*. A interface `IStreamMessage` contém os métodos para processar o corpo de uma mensagem de fluxo. Para mais informações, consulte [“Mensagens de Fluxo” na página 80](#).

#### text

O corpo contém uma sequência. Uma mensagem com esse tipo de corpo é chamada de *mensagem de texto*. A interface `ITextMessage` contém os métodos para processar o corpo de uma mensagem de texto. Para mais informações, consulte [“Mensagens de texto” na página 81](#).

A interface IMessage é o pai de todos os objetos de mensagem e pode ser usada em funções do sistema de mensagens para representar qualquer um dos tipos de mensagens do XMS.

Para obter informações sobre o tamanho e o máximo e o mínimo de valores de cada um desses tipos de dados, consulte [Tabela 5 na página 40](#).

### Referências relacionadas

[Partes de uma mensagem XMS](#)

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

[Campos de cabeçalho em uma mensagem XMS](#)

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS, o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS.

[Propriedades de uma mensagem XMS](#)

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

[Seletores de mensagens](#)

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

[Mapeando mensagens do XMS para mensagens do IBM MQ](#)

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

Por esse motivo, cada elemento de dados do aplicativo gravado no corpo de uma mensagem por um aplicativo XMS deve ter um dos tipos de dados listados em [Tabela 21 na página 77](#). Para cada tipo de dados, a tabela mostra o tipo de dados Java compatível. XMS fornece os métodos para gravar elementos de dados do aplicativo apenas com esses tipos de dados.

<b>XMSTipo de Dados</b>	<b>Representa</b>	<b>Tipo de DadosJava Compatível</b>
System.Boolean	O valor booleano true ou false	booleano
System.Char16	Caractere de byte duplo	caractere
System.SByte	Inteiro de 8 bits sinalizado	byte
System.Int16	Número inteiro de 16 bits assinado	short
System.Int32	Número inteiro de 32 bits assinado	int
System.Int64	Número inteiro de 64 bits sinalizado	grande
System.Float	Número de vírgula flutuante assinado	float
System.Double	Número de vírgula flutuante de precisão dupla assinado	duplo
System.String	Sequência de caracteres	Sequência

Para obter informações sobre o tamanho, o valor máximo e o valor mínimo de cada um desses tipos de dados, consulte [“Tipos primitivos do XMS” na página 40](#).

### Conceitos relacionados

[Atributos e Propriedades de Objetos](#)

Um objeto XMS pode ter atributos e propriedades, que são características do objeto, que são implementados de diferentes maneiras

### Tipos primitivos do XMS

O XMS fornece equivalentes dos oito tipos primitivos Java (byte, short, int, long, float, double, char e boolean). Isso permite a troca de mensagens entre o XMS e o JMS sem que os dados se tornem perdidos ou corrompidos.

### Conversão implícita de um valor de propriedade de um tipo de dados para outro

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

### **Referências relacionadas**

#### Mensagens de Bytes

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

#### Mensagens de Mapa

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

#### Mensagens de objeto

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

#### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### Mensagens de texto

O corpo de uma mensagem de texto contém uma sequência.

### ***Mensagens de Bytes***

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

As mensagens de bytes são úteis se um aplicativo XMS precisa trocar mensagens com aplicativos que não estão usando a interface de programação de aplicativos XMS ou JMS.

Depois que um aplicativo cria uma mensagem de byte, o corpo da mensagem é somente gravação. O aplicativo monta os dados do aplicativo para o corpo, chamando os métodos de gravação apropriados da interface `IBytesMessage` para .NET. Toda vez que o aplicativo grava um valor para o fluxo de mensagens de bytes, o valor é montado imediatamente após o valor anterior gravado pelo aplicativo. XMS mantém um cursor interno para lembrar a posição do último byte que foi montado.

Quando o aplicativo envia a mensagem, o corpo da mensagem se torna somente leitura. Nesse modo, o aplicativo pode enviar a mensagem repetidamente.

Quando um aplicativo recebe uma mensagem de bytes, o corpo da mensagem é de leitura. O aplicativo pode usar os métodos de leitura apropriados da interface `IBytesMessage` para ler o conteúdo do fluxo de mensagens de bytes. O aplicativo lê os bytes na sequência e XMS mantém um cursor interno para lembrar a posição do último byte que foi lido.

Se um aplicativo chamar o método `Reconfigurar` da interface `IBytesMessage` quando o corpo de uma mensagem de bytes for gravável, o corpo se tornará somente leitura. O método também reposiciona o cursor no início do fluxo de mensagens de bytes.

Se um aplicativo chamar o método `Clear Body` da interface `IMessage` para o .NET quando o corpo de uma mensagem de bytes for somente leitura, o corpo se tornará gravável. O método também limpa o corpo.

### **Referências relacionadas**

#### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

#### Mensagens de Mapa

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

#### Mensagens de objeto

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

#### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### Mensagens de texto

O corpo de uma mensagem de texto contém uma sequência.

#### IBytesMessage (para a interface .NET)

Uma mensagem de bytes é uma mensagem cujo corpo compreende um fluxo de bytes.

### ***Mensagens de Mapa***

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

Em cada par de nome-valor, o nome é uma sequência que identifica o valor e o valor é um elemento de dados do aplicativo que possui um dos tipos de dados XMS listados em [Tabela 21 na página 77](#). A ordem dos pares nome-valor não está definida. A classe MapMessage contém os métodos para configurar e obter pares nome-valor.

Um aplicativo pode acessar um par nome-valor aleatoriamente, especificando seu nome.

Um aplicativo .NET pode usar a propriedade MapNames para obter uma enumeração dos nomes no corpo da mensagem de mapa.

Quando um aplicativo obtém o valor de um par nome-valor, o valor pode ser convertido por XMS em outro tipo de dados. Por exemplo, para obter um número inteiro a partir do corpo de uma mensagem de mapa, um aplicativo pode chamar o método GetString da classe MapMessage, que retorna o número inteiro como uma sequência. As conversões suportadas são as mesmas que aquelas suportadas quando XMS converte um valor de propriedade de um tipo de dados para outro. Para obter mais informações sobre as conversões suportadas, consulte [“Conversão implícita de um valor de propriedade de um tipo de dados para outro” na página 41](#).

Depois que um aplicativo cria uma mensagem de mapa, o corpo da mensagem é legível e gravável. O corpo permanece legível e gravável depois que o aplicativo envia a mensagem. Quando um aplicativo recebe uma mensagem de mapa, o corpo da mensagem é de leitura. Se um aplicativo chamar o método Clear Body da classe de mensagem quando o corpo de uma mensagem de mapa for de leitura, o corpo se tornará legível e gravável. O método também limpa o corpo.

#### **Conceitos relacionados**

##### Conversão implícita de um valor de propriedade de um tipo de dados para outro

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

#### **Referências relacionadas**

##### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

##### Mensagens de Bytes

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

##### Mensagens de objeto

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

##### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### Mensagens de texto

O corpo de uma mensagem de texto contém uma sequência.

#### IMapMessage (para a interface .NET)

Uma mensagem de mapa é uma mensagem cujo corpo consiste em um conjunto de pares nome-valor, em que cada valor possui um tipo de dados associado.

### **Mensagens de objeto**

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

Um aplicativo XMS pode receber uma mensagem de objeto, alterar seus campos de cabeçalho e propriedades e, em seguida, enviá-lo para outro destino. Um aplicativo também pode copiar o corpo de uma mensagem de objeto e usá-lo para formar outra mensagem de objeto. XMS trata o corpo de uma mensagem de objeto como uma matriz de bytes.

Depois que um aplicativo cria uma mensagem de objeto, o corpo da mensagem é legível e gravável. O corpo permanece legível e gravável depois que o aplicativo envia a mensagem. Quando um aplicativo recebe uma mensagem de objeto, o corpo da mensagem é de leitura. Se um aplicativo chamar o método `Clear Body` da interface de `IMessage` para .NET quando o corpo de uma mensagem de objeto for de leitura, o corpo se tornará legível e gravável. O método também limpa o corpo.

### **Referências relacionadas**

#### Tipos de Dados para Elementos de Dados do Aplicativo

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

#### Mensagens de Bytes

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

#### Mensagens de Mapa

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

#### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### Mensagens de texto

O corpo de uma mensagem de texto contém uma sequência.

#### IObjectMessage (para a interface .NET)

Uma mensagem de objeto é uma mensagem cujo corpo compreende um objeto Java ou .NET serializado.

### **Mensagens de Fluxo**

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

O tipo de dados de um valor é um dos tipos de dados XMS listados em [Tabela 21 na página 77](#).

Depois que um aplicativo cria uma mensagem de fluxo, o corpo da mensagem é gravável. O aplicativo monta os dados do aplicativo para o corpo, chamando os métodos de gravação apropriados da interface `IStreamMessage` para .NET. Toda vez que o aplicativo grava um valor para o fluxo de mensagens, o valor e seu tipo de dados são montados imediatamente após o valor anterior gravado pelo aplicativo. XMS mantém um cursor interno para lembrar a posição do último valor que foi montado.

Quando o aplicativo envia a mensagem, o corpo da mensagem se torna somente leitura. Nesse modo, o aplicativo pode enviar a mensagem várias vezes.

Quando um aplicativo recebe uma mensagem de fluxo, o corpo da mensagem é de leitura. O aplicativo pode usar os métodos de leitura apropriados da interface `IStreamMessage` para .NET para ler o conteúdo

do fluxo de mensagens. O aplicativo lê os valores na sequência, e XMS mantém um cursor interno para lembrar a posição do último valor que foi lido.

Quando um aplicativo lê um valor do fluxo de mensagens, o valor pode ser convertido pelo XMS em outro tipo de dados. Por exemplo, para ler um número inteiro a partir do fluxo de mensagens, um aplicativo pode chamar o método `ReadString`, que retorna o número inteiro como uma sequência. As conversões suportadas são as mesmas que aquelas suportadas quando XMS converte um valor de propriedade de um tipo de dados para outro. Para obter mais informações sobre as conversões suportadas, consulte [“Conversão implícita de um valor de propriedade de um tipo de dados para outro”](#) na página 41.

Se um erro ocorrer enquanto um aplicativo está tentando ler um valor do fluxo de mensagens, o cursor não está avançado. O aplicativo pode se recuperar do erro ao tentar ler o valor como outro tipo de dados.

Se um aplicativo chamar o método `Reconfigurar` da interface `IStreamMessage` para XMS quando o corpo de uma mensagem de fluxo for de gravação somente, o corpo se tornará somente leitura. O método também reposiciona o cursor no início do fluxo de mensagens.

Se um aplicativo chamar o método `Clear Body` da interface do `IMessage` para XMS quando o corpo de uma mensagem de fluxo for somente leitura, o corpo se tornará somente gravação. O método também limpa o corpo.

### **Conceitos relacionados**

[Conversão implícita de um valor de propriedade de um tipo de dados para outro](#)

Quando um aplicativo obtém o valor de uma propriedade, o valor pode ser convertido por XMS em outro tipo de dados. Muitas regras controlam quais conversões são suportadas e como o XMS executa as conversões.

### **Referências relacionadas**

[Tipos de Dados para Elementos de Dados do Aplicativo](#)

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

[Mensagens de Bytes](#)

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

[Mensagens de Mapa](#)

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

[Mensagens de objeto](#)

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

[Mensagens de texto](#)

O corpo de uma mensagem de texto contém uma sequência.

[IStreamMessage \(para a interface .NET\)](#)

Uma mensagem de fluxo é uma mensagem cujo corpo compreende um fluxo de valores, em que cada valor possui um tipo de dados associado. Os conteúdos do corpo são gravados e lidos sequencialmente.

### ***Mensagens de texto***

O corpo de uma mensagem de texto contém uma sequência.

Depois que um aplicativo cria uma mensagem de texto, o corpo da mensagem é legível e gravável. O corpo permanece legível e gravável depois que o aplicativo envia a mensagem. Quando um aplicativo recebe uma mensagem de texto, o corpo da mensagem é somente leitura. Se um aplicativo chamar o método `Clear Body` da interface `IMessage` para .NET quando o corpo de uma mensagem de texto for de leitura, o corpo se tornará legível e gravável. O método também limpa o corpo.

### **Referências relacionadas**

[Tipos de Dados para Elementos de Dados do Aplicativo](#)

Para assegurar que um aplicativo XMS possa trocar mensagens com um aplicativo IBM MQ classes for JMS, ambos os aplicativos devem ser capazes de interpretar os dados do aplicativo no corpo de uma mensagem da mesma maneira.

#### Mensagens de Bytes

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

#### Mensagens de Mapa

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

#### Mensagens de objeto

O corpo de uma mensagem de objeto contém um objeto serializado Java ou .NET .

#### Mensagens de Fluxo

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

#### ITextMessage (para a interface .NET)

Uma mensagem de texto é uma mensagem cujo corpo compreende uma sequência.

## **Seletores de mensagens**

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

Quando um aplicativo cria um consumidor de mensagens, ele pode associar uma expressão de seletor de mensagem ao consumidor. A expressão do seletor de mensagem especifica os critérios de seleção.

Quando um aplicativo está se conectando ao gerenciador de filas IBM WebSphere MQ 7.0 , a seleção de mensagens é feita no lado do gerenciador de filas. XMS não faz nenhuma seleção e simplesmente entrega a mensagem que recebeu do gerenciador de filas, fornecendo, assim, melhor desempenho.

Um aplicativo pode criar mais de um consumidor de mensagens, cada um com sua própria expressão de seletor de mensagem. Se uma mensagem recebida atender aos critérios de seleção de mais de um consumidor de mensagens, o XMS entregará a mensagem para cada um desses consumidores.

Uma expressão de seletor de mensagem pode referenciar as seguintes propriedades de uma mensagem:

- Propriedades Definidas pelo JMS
- Propriedades Definidas pela IBM
- Propriedades Definidas pelo Aplicativo

Ele também pode referenciar os campos de cabeçalho da mensagem a seguir:

- JMSCorrelationID
- JMSDeliveryMode
- JMSMessageID
- JMSPriority
- JMSTimestamp
- JMSType

Uma expressão de seletor de mensagem, no entanto, não pode referenciar dados no corpo de uma mensagem.

Aqui está um exemplo de uma expressão de seletor de mensagem:

```
JMSPriority > 3 AND manufacturer = 'Jaguar' AND model in ('xj6','xj12')
```

O XMS entrega uma mensagem para um consumidor de mensagens com essa expressão de seletor de mensagem apenas se a mensagem tiver uma prioridade maior que 3; uma propriedade definida pelo aplicativo, fabricante, com um valor de Jaguar; e outra propriedade definida pelo aplicativo, modelo, com um valor de xj6 ou xj12.

As regras de sintaxe para formar uma expressão de seletor de mensagem em XMS são as mesmas que aquelas em IBM MQ classes for JMS. Para obter informações sobre como construir uma expressão do seletor de mensagens, consulte a Nota de documentação do produto IBM MQ que, em uma expressão do seletor de mensagens, os nomes de propriedades definidas de JMS devem ser os nomes de JMS, e os de propriedades definidas de IBM devem ser os nomes IBM MQ classes for JMS. Não é possível usar os nomes XMS em uma expressão de seletor de mensagem.

### Referências relacionadas

#### Partes de uma mensagem XMS

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

#### Campos de cabeçalho em uma mensagem XMS

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS , o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS .

#### Propriedades de uma mensagem XMS

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

#### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

#### Mapeando mensagens do XMS para mensagens do IBM MQ

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

## Mapeando mensagens do XMS para mensagens do IBM MQ

Os campos de cabeçalho JMS e propriedades de uma mensagem XMS são mapeados em campos nas estruturas de cabeçalho de uma mensagem IBM MQ.

Quando um aplicativo XMS está conectado a um gerenciador de filas IBM MQ , as mensagens enviadas para o gerenciador de filas são mapeadas para as mensagens IBM MQ da mesma maneira que as mensagens IBM MQ classes for JMS são mapeadas para mensagens IBM MQ em circunstâncias semelhantes.

Se a propriedade `XMSC_WMQ_TARGET_CLIENT` de um objeto Destination estiver configurada como `XMSC_WMQ_TARGET_DEST_JMS`, os campos de cabeçalho JMS e as propriedades de uma mensagem enviada para o destino serão mapeados para os campos nas estruturas de cabeçalho MQMD e MQRFH2 da mensagem IBM MQ . Configurar a propriedade `XMSC_WMQ_TARGET_CLIENT` dessa maneira assume que o aplicativo que recebe a mensagem pode manipular um cabeçalho MQRFH2. O aplicativo de recebimento pode, portanto, ser outro aplicativo XMS , um aplicativo IBM MQ classes for JMS ou um aplicativo IBM MQ nativo que tenha sido projetado para manipular um cabeçalho MQRFH2.

Se a propriedade `XMSC_WMQ_TARGET_CLIENT` de um objeto de Destino for configurada como `XMSC_WMQ_TARGET_DEST_MQ` em vez disso, os campos de cabeçalho JMS e as propriedades de uma mensagem enviada para o destino serão mapeados para os campos na estrutura de cabeçalho MQMD da mensagem IBM MQ . A mensagem não contém um cabeçalho MQRFH2 e quaisquer campos de cabeçalho JMS e propriedades que não podem ser mapeados para campos na estrutura de cabeçalho MQMD são ignorados. O aplicativo que recebe a mensagem pode, portanto, ser um IBM MQ nativo que não foi projetado para manipular um cabeçalho MQRFH2.

As mensagens do IBM MQ recebidas de um gerenciador de filas são mapeadas para mensagens do XMS da mesma maneira que mensagens do IBM MQ são mapeadas para mensagens do IBM MQ classes for JMS em circunstâncias semelhantes.

Se uma mensagem IBM MQ recebida tiver um cabeçalho MQRFH2, a mensagem XMS resultante terá um corpo cujo tipo seja determinado pelo valor da propriedade **Ms**d contida na pasta mcd do cabeçalho MQRFH2. Se a propriedade **Ms**d não estiver presente no cabeçalho MQRFH2, ou se a mensagem IBM MQ não tiver cabeçalho MQRFH2, a mensagem XMS resultante terá um corpo cujo tipo seja determinado pelo valor do campo *Format* no cabeçalho MQMD. Se o campo *Format* estiver configurado como `MQFMT_STRING`, a mensagem XMS será uma mensagem de texto. Caso contrário, a mensagem XMS é

uma mensagem de bytes. Se a mensagem IBM MQ não tiver cabeçalho MQRFH2, apenas os campos de cabeçalho JMS e propriedades que podem ser derivados de campos no cabeçalho MQMD são configurados.

Para obter mais informações sobre o mapeamento de IBM MQ classes for JMS mensagens para IBM MQ mensagens, consulte a documentação do produto IBM MQ .

### **Referências relacionadas**

#### Partes de uma mensagem XMS

Uma mensagem XMS consiste em um cabeçalho, um conjunto de propriedades e um corpo.

#### Campos de cabeçalho em uma mensagem XMS

Para permitir que um aplicativo XMS troque mensagens com um aplicativo WebSphereJMS , o cabeçalho de uma mensagem XMS contém os campos de cabeçalho da mensagem JMS .

#### Propriedades de uma mensagem XMS

XMS suporta três tipos de propriedade de mensagem: JMS propriedades definidas, propriedades definidas do IBM e propriedades definidas pelo aplicativo.

#### O corpo de uma mensagem XMS

O corpo de uma mensagem contém dados do aplicativo. No entanto, uma mensagem não pode ter nenhum corpo e engloba apenas os campos de cabeçalho e propriedades.

#### Seletores de mensagens

Um aplicativo XMS usa seletores de mensagens para selecionar as mensagens que ele deseja receber.

### ***Lendo e gravando o descritor de mensagens a partir de um aplicativo IBM Message Service Client for .NET***

É possível acessar todos os campos do descritor de mensagens (MQMD) de uma mensagem IBM MQ , exceto StrucId e Versão; BackoutCount pode ser lido, mas não gravado. Esse recurso está disponível apenas ao conectar-se a um gerenciador de filas do IBM WebSphere MQ 6.0 ou mais recente e é controlado pelas propriedades de destino descritas posteriormente.

Os atributos da mensagem fornecidos pelo IBM Message Service Client for .NET facilitam os aplicativos XMS para configurar campos MQMD e também para a unidade de aplicativos IBM WebSphere MQ .

Algumas restrições se aplicam ao usar o sistema de mensagens de publicação / assinatura. Por exemplo, os campos MQMD como MsgID e CorrelId, se configurados, são ignorados.

A função descrita neste tópico está indisponível para o sistema de mensagens de publicação/assinatura ao conectar-se com um gerenciador de filas do IBM WebSphere MQ 6.0. Ele também estará indisponível quando a propriedade **PROVIDERVERSION** for configurada como 6.

### ***Acessando os dados da mensagem IBM MQ de um aplicativo IBM Message Service Client for .NET***

Você pode acessar os dados da mensagem completos IBM MQ, incluindo o cabeçalho MQRFH2 (se presente) e quaisquer outros cabeçalhos IBM MQ (se presentes) dentro de um aplicativo IBM Message Service Client for .NET como o corpo de uma JMSBytesMessage.

A função descrita neste tópico está disponível apenas ao conectar-se com um gerenciador de filas do IBM WebSphere MQ 7.0 ou mais recente e quando o provedor de sistemas de mensagens do WebSphere MQ está no modo normal.

As propriedades do objeto de destino determinam como o aplicativo XMS acessa o conjunto de uma mensagem IBM MQ (incluindo o cabeçalho MQRFH2, se presente) como o corpo de uma JMSBytesMessage.

## **Resolução de Problemas**

---

Esta seção fornece informações para ajudá-lo a detectar e lidar com problemas ao usar IBM Message Service Client for .NET.

Esta seção contém os seguintes tópicos:

- [“Configuração de rastreo para aplicativos .NET” na página 85](#)
- [“Configuração de FFDC para aplicativos .NET” na página 89](#)
- [“Dicas para Resolução de Problemas” na página 89](#)

## Configuração de rastreo para aplicativos .NET

Para aplicativos XMS .NET, é possível configurar o rastreo por meio de um arquivo de configuração do aplicativo, bem como por meio das variáveis de ambiente do XMS. É possível selecionar os componentes que você deseja rastrear. O rastreo é normalmente usado sob a orientação do Suporte IBM.

O rastreo para o XMS .NET baseia-se na infraestrutura de rastreo padrão do .NET.

Todo o rastreo, exceto o rastreo de erro, fica desativado por padrão. É possível ativar o rastreo e definir as configurações de rastreo de uma das maneiras a seguir:

- Usando um arquivo de configuração de aplicativo com um nome que consiste no nome do programa executável ao qual o arquivo se relaciona, com o sufixo `.config`. Por exemplo, o arquivo de configuração de aplicativo para `text.exe` teria o nome `text.exe.config`. O uso de um arquivo de configuração de aplicativo é a maneira preferida de ativar o rastreo para aplicativos XMS.NET. Para obter detalhes adicionais, consulte [“Configuração de rastreo usando um arquivo de configuração de aplicativo” na página 86](#).
- Usando as variáveis de ambiente XMS como para aplicativos XMS C ou C++. Para obter detalhes adicionais, consulte [“Configuração de rastreo usando variáveis de ambiente XMS” na página 88](#).

O arquivo de rastreo ativo possui um nome do formato `xms_tracePID.log` em que `PID` representa o ID do processo do aplicativo. O tamanho do arquivo de rastreo ativo está, por padrão, limitado a 20 MB. Quando esse limite é atingido, o arquivo é renomeado e arquivado. Os arquivos arquivados possuem nomes do formato `xms_tracePID_YY.MM.DD_HH.MM.SS.log`

Por padrão, o número de arquivos de rastreo retidos é quatro, isto é, um arquivo ativo e três arquivos arquivados. Esses quatro arquivos são usados como um buffer de rolagem até que o aplicativo pare, com o arquivo mais antigo sendo removido e substituído pelo arquivo mais recente. É possível mudar o número de arquivos de rastreo especificando um número diferente no arquivo de configuração de aplicativo. No entanto, deve haver pelo menos dois arquivos (um arquivo ativo e um arquivo arquivado).

Há dois formatos de arquivo de rastreo disponíveis:

- Os arquivos de rastreo de formato básico são legíveis por humanos em um formato WebSphere Application Server. Esse é o formato de arquivo de rastreo padrão. O formato básico não é compatível com as ferramentas do analisador de rastreo.
- Os arquivos de rastreo no formato avançado são compatíveis com as ferramentas do analisador de rastreo. Deve-se especificar o desejo de produzir arquivos de rastreo no formato avançado no arquivo de configuração de aplicativo.

As entradas de rastreo contêm as informações a seguir:

- A data e hora em que o rastreo foi registrado
- O nome de classe
- O tipo de rastreo
- A mensagem de rastreo

O exemplo a seguir mostra uma extração de algum 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
```

No exemplo anterior, o formato é:

[Date Time:Microsecs] or Exit	Thread-id	Classname	Trace-type	Methodname	Entry
----------------------------------	-----------	-----------	------------	------------	-------

em que Trace-type é:

- > para entrada
- < para saída
- d para informações sobre depuração

## Configuração de rastreo usando um arquivo de configuração de aplicativo

A maneira preferida de configurar o rastreo para aplicativos XMS .NET é com um arquivo de configuração de aplicativo. A seção de rastreo desse arquivo inclui parâmetros que definem o que deve ser rastreado, o local do arquivo de rastreo e o tamanho máximo permitido, o número de arquivos de rastreo usados e o formato do arquivo de rastreo.

Para ativar o rastreo usando o arquivo de configuração de aplicativo, simplesmente coloque o arquivo no mesmo diretório que o arquivo executável para seu aplicativo.

O rastreo pode ser ativado por componente e tipo de rastreo. Também é possível ativar o rastreo para um grupo inteiro de rastreo. É possível ativar o rastreo para componentes em uma hierarquia, individual ou coletivamente. Os tipos de rastreo disponíveis incluem:

- Rastreamento de Depuração
- Rastreo da Exceção
- Avisos, mensagens informativas e mensagens de erro
- Rastreo de entrada e saída do método

O exemplo a seguir mostra as configurações de rastreo definidas na seção Rastreo de um arquivo de configuração de aplicativo:

```
<?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>
```

[Tabela 22 na página 87](#) descreve as configurações de parâmetros em mais detalhes.

Tabela 22. Configurações de parâmetros de rastreo do arquivo de configuração de aplicativo

Parâmetro	Descrição
<code>traceSpecification=ComponentName=type=state</code>	<p><i>ComponentName</i> é o nome da classe que você deseja rastrear. É possível usar um caractere curinga * neste nome. Por exemplo, <code>*=all=enabled</code> especifica que você deseja rastrear todas as classes e <code>IBM.XMS.impl.*=all=enabled</code> especifica que é necessário apenas o rastreo de API.</p> <p><i>type</i> pode ser qualquer um dos seguintes tipos de rastreo:</p> <ul style="list-style-type: none"> <li>• all</li> <li>• debug</li> <li>• evento</li> <li>• EntryExit</li> </ul> <p><i>state</i> pode ser ativado ou desativado.</p> <p>É possível sequenciar vários elementos de rastreo juntos usando um delimitador ':' (dois pontos).</p>
<code>traceFilePath="filename"</code>	<p>Se você não especificar um <code>traceFilePath</code> ou se o <code>traceFilePath</code> estiver presente, mas contiver uma sequência vazia, o arquivo de rastreo será colocado no diretório atual. Para armazenar o arquivo de rastreo em um diretório nomeado, especifique o nome de diretório no <code>traceFilePath</code>, por exemplo:</p> <pre>traceFilePath="c:\somepath"</pre>
<code>traceFileSize="size"</code>	<p>O tamanho máximo permitido do arquivo de rastreo. Quando um arquivo atinge este tamanho, ele é arquivado e renomeado. O máximo padrão é 20 KB, especificado como <code>traceFileSize="20000000"</code>.</p>
<code>traceFileNumber="number"</code>	<p>O número de arquivos de rastreo que devem ser retidos. O padrão é 4 (um arquivo ativo e 3 archives). O número mínimo permitido é 2.</p>
<code>traceFormat="format"</code>	<p>O formato de rastreo padrão é básico. Os arquivos de rastreo serão produzidos nesse formato se você especificar <code>traceFormat="basic"</code>, se não especificar um <code>traceFormat</code> ou se o <code>traceFormat</code> estiver presente, mas contiver uma sequência vazia.</p> <p>Caso seja necessário um rastreo compatível com as ferramentas do analisador de rastreo, deve-se especificar <code>traceFormat="advanced"</code>.</p>

As configurações de rastreo no arquivo de configuração de aplicativo são dinâmicas e lidas novamente sempre que o arquivo é salvo ou substituído. Se forem localizados erros no arquivo após sua edição, as configurações do arquivo de rastreo serão revertidas para seus valores padrão.

#### Conceitos relacionados

[Configuração de rastreo usando variáveis de ambiente XMS](#)

Como alternativa ao uso de um arquivo de configuração do aplicativo, é possível ativar o rastreo usando variáveis de ambiente do XMS. Essas variáveis de ambiente serão usadas apenas se não houver especificação de rastreo no arquivo de configuração do aplicativo.

## Configuração de rastreo usando variáveis de ambiente XMS

Como alternativa ao uso de um arquivo de configuração do aplicativo, é possível ativar o rastreo usando variáveis de ambiente do XMS. Essas variáveis de ambiente serão usadas apenas se não houver especificação de rastreo no arquivo de configuração do aplicativo.

Para configurar o rastreo de um aplicativo XMS .NET, configure as variáveis de ambiente a seguir, antes de executar o aplicativo:

<i>Tabela 23. Configurações da variável de ambiente para o rastreo .NET</i>			
<b>Variáveis de ambiente</b>	<b>Padrão</b>	<b>Configurações</b>	<b>Significado</b>
XMS_TRACE_ON	Não-aplicável	Não aplicável: o valor dessa variável é ignorado	Se XMS_TRACE_ON estiver configurado, todo o rastreo será ativado por padrão.
XMS_TRACE_FILE_PATH	Diretório de trabalho atual	/dirpath/	<p>O caminho do diretório no qual os registros de rastreo e de FFDC são gravados.</p> <p>O XMS cria arquivos FFDC e de rastreo no diretório ativo atual, a menos que você especifique um local alternativo. É possível especificar um local alternativo configurando a variável de ambiente XMS_TRACE_FILE_PATH para o nome do caminho completo do diretório em que você deseja que o XMS crie os arquivos FFDC e de rastreo. Deve-se configurar essa variável de ambiente antes de iniciar o aplicativo que você deseja rastrear. Deve-se certificar de que o identificador de usuário sob o qual o aplicativo é executado tenha a autoridade para gravar no diretório em que o XMS cria os arquivos FFDC e de rastreo.</p>

Tabela 23. Configurações da variável de ambiente para o rastreo .NET (continuação)

Variáveis de ambiente	Padrão	Configurações	Significado
XMS_TRACE_FORMAT	BÁSICA	BASIC, ADVANCED	Especifica o formato de rastreo necessário, que pode ser BASIC ou ADVANCED. O formato padrão é BASIC. O formato ADVANCED é compatível com as ferramentas do analisador de rastreo.
XMS_TRACE_SPECIFICATION	Não-aplicável	Consulte <a href="#">“Configuração de rastreo usando um arquivo de configuração de aplicativo”</a> na página 86	Substitui a especificação de rastreo, que segue o formato especificado em <a href="#">“Configuração de rastreo usando um arquivo de configuração de aplicativo”</a> na página 86.

### Conceitos relacionados

#### [Configuração de rastreo usando um arquivo de configuração de aplicativo](#)

A maneira preferida de configurar o rastreo para aplicativos XMS.NET é com um arquivo de configuração de aplicativo. A seção de rastreo desse arquivo inclui parâmetros que definem o que deve ser rastreado, o local do arquivo de rastreo e o tamanho máximo permitido, o número de arquivos de rastreo usados e o formato do arquivo de rastreo.

## Configuração de FFDC para aplicativos .NET

Para a implementação .NET do XMS, um arquivo FFDC é produzido para cada FFDC.

Os arquivos de Primeira captura de dados com falha (FFDC) são armazenados em arquivos de texto legíveis por humanos. Os nomes desses arquivos têm o formato `xmsffdcprocessID_DateTimestamp.txt`. Um exemplo de um nome de arquivo é `xmsffdc264_2006.01.06T13.18.52.990955.txt`. O registro de data e hora contém a resolução de microssegundos.

Os arquivos começam com a data e hora em que a exceção ocorreu, seguido pelo tipo de exceção. Os arquivos incluem um probeId curto exclusivo, que pode ser usado para localizar onde esse FFDC ocorreu.

Não é necessário realizar nenhuma configuração para ativar o FFDC. Por padrão, todos os arquivos FFDC são gravados no diretório atual. No entanto, se necessário, é possível especificar um diretório diferente mudando `ffdcDirectory` na seção Rastreo do arquivo de configuração de aplicativo. No exemplo a seguir, todos os arquivos de rastreo são registrados no diretório `c:\client\ffdc`:

```
<IBM.XMS>
  <Trace ffdc=true ffdcDirectory="c:\client\ffdc"/>
</IBM.XMS>
```

É possível desativar o rastreo configurando o FFDC como `false` na seção Rastreo do arquivo de configuração do aplicativo.

Se você não estiver usando um arquivo de configuração do aplicativo, o FFDC estará ativo e o rastreo desativado.

## Dicas para Resolução de Problemas

Use estas dicas para ajudar a solucionar problemas com o uso do XMS.

## **Um aplicativo XMS não pode se conectar a um gerenciador de filas (MQRC\_NOT\_AUTHORIZED)**

O cliente XMS .NET pode ter um comportamento diferente do comportamento do cliente IBM MQ JMS. Portanto, é possível perceber que seu aplicativo XMS não pode se conectar a seu gerenciador de filas, embora seu aplicativo JMS possa.

- Uma solução simples para esse problema é tentar usar um ID de usuário que não tenha mais de 12 caracteres de comprimento e esteja autorizado completamente na lista de autoridade do gerenciador de filas. Se essa solução não for ideal, uma abordagem diferente, mas mais complexa, será a utilização de saídas de segurança. Se você precisar de ajuda adicional sobre esse problema, entre em contato com o Suporte IBM para obter assistência.
- Se você configurar a propriedade XMSC\_USERID do connection factory, ela deverá corresponder ao ID de usuário e à senha do usuário conectado. Se você não configurar essa propriedade, o gerenciador de filas usará o ID de usuário do usuário conectado, por padrão.
- A autenticação do usuário para o IBM MQ é executada usando os detalhes do usuário atualmente conectado e não as informações fornecidas nos campos XMSC.USERID e XMSC.PASSWORD. Isso foi projetado para manter a consistência com o IBM MQ. Para obter mais informações sobre autenticação, consulte *Informações sobre autenticação* na documentação do produto IBM MQ on-line.

## **Conexão redirecionada para o mecanismo do sistema de mensagens**

Ao conectar-se com um barramento de integração de serviços do WebSphere Application Server 6.0.2, todas as conexões podem ser redirecionadas do terminal do provedor original para o mecanismo do sistema de mensagens escolhido pelo barramento para essa conexão do cliente. Ao fazer isso, ele sempre redirecionará a conexão para um servidor host especificado pelo nome do host, em vez de por um endereço IP. Portanto, você poderá ter problemas de conexão se o nome do host não puder ser resolvido.

Para se conectar com sucesso ao barramento de integração de serviços do WebSphere Application Server 6.0.2, talvez seja necessário fornecer um mapeamento entre os nomes de host e os endereços IP em sua máquina host do cliente. Por exemplo, é possível especificar o mapeamento em uma tabela de hosts locais em sua máquina host do cliente.

## **Suporte para autenticação de senha semelhante a telnet**

O protocolo XMS .NET Real Time Transport suporta apenas autenticação de senha semelhante a telnet simples. O protocolo XMS .NET Real Time Transport não suporta Qualidade de proteção.

## **Configurando valores para o tipo de propriedade double**

Em uma plataforma Windows de 64 bits, os métodos SetDoubleProperty() ou GetDoubleProperty() poderão não funcionar corretamente ao configurar ou obter valores para o tipo de propriedade double, se os valores forem menores que Double.Epsilon.

Por exemplo, se você tentar configurar um valor de 4.9E-324 para uma propriedade com o tipo double, as plataformas Windows de 64 bits tratarão como 0.0. Portanto, em um ambiente de sistema de mensagens distribuído, se um JMS ou outro aplicativo configurar o valor para uma propriedade double como 4.9E-324 em qualquer máquina UNIX ou Windows de 32 bits e o XMS.NET for executado em uma máquina de 64 bits, o valor retornado por GetDoubleProperty() será 0.0. Esse é um problema conhecido com o Microsoft .NET Framework 2.0 Framework.

## **Condições de erro que podem ser manipuladas no tempo de execução**

Os códigos de retorno de chamadas de API são condições de erro que podem ser manipulados no tempo de execução. A maneira na qual você lida com esse tipo de erro depende se você está usando a API C ou C++.

## Como detectar erros no tempo de execução

Se um aplicativo chamar uma função da API C e a chamada falhar, uma resposta com um código de retorno diferente de XMS\_OK será retornado com um bloco de erro XMS contendo mais informações sobre o motivo da falha.

A API C++ lança uma exceção quando um método é usado..

Um aplicativo usa um listener de exceção para ser notificado assincronamente de um problema com uma conexão. O listener de exceção é fornecido e inicializado usando a API XMS C ou C++ .

## Como manipular erros no tempo de execução

Algumas condições de erro são uma indicação de que algum recurso está indisponível, e a ação que um aplicativo pode executar depende da função XMS que o aplicativo está chamando.. Por exemplo, se uma conexão falhar ao se conectar ao servidor, o aplicativo poderá desejar tentar novamente periodicamente até que uma conexão seja feita Um bloco ou exceção de erro XMS pode não conter informações suficientes para determinar qual ação tomar e, nessas situações, geralmente há um bloco ou exceção de erro vinculado que contém informações de diagnóstico mais específicas.

Na API C, sempre teste uma resposta com um código de retorno diferente de XMS\_OK e sempre transmita um bloco de erro na chamada da API. A ação tomada geralmente depende de qual função da API é o aplicativo usando.

Na API C++ , sempre inclua chamadas para métodos em um bloco try e, para capturar todos os tipos de exceção XMS , especifique a classe Exception na construção catch.

O listener de exceção é um caminho de condição de erro assíncrono que pode ser iniciado a qualquer momento.. Quando a função do listener de exceção é iniciada, em seu próprio encadeamento, ela geralmente é uma indicação de uma falha mais grave do que uma condição de erro normal da API do XMS . Qualquer ação apropriada pode ser executada mas deve-se ter cuidado para seguir as regras para o modelo de encadeamento XMS , conforme descrito em [“O modelo de encadeamento” na página 21](#)

### Conceitos relacionados

[O modelo de encadeamento](#)

As regras gerais controlam como um aplicativo multiencadeado pode usar objetos XMS .

## Cientes de serviço de mensagens para referência .NET

Esta seção de referência fornece informações para ajudá-lo a usar Message Service Client for .NET. Essas informações ajudam você a executar as tarefas envolvidas na programação com o XMS

## .REDE interfaces

Esta seção documenta as interface de classe .REDE e suas propriedades e métodos.

A tabela a seguir resume todas as interfaces, que são definidas no namespace IBM . XMS .

Interface	Descrição
<a href="#">“IBytesMessage” na página 94</a>	Uma mensagem de bytes é uma mensagem cujo corpo compreende um fluxo de bytes.
<a href="#">“IConnection” na página 104</a>	Um objeto Connection representa a conexão ativa do aplicativo para um servidor de sistema de mensagens.
<a href="#">“IConnectionFactory” na página 107</a>	Um aplicativo usa um connection factory para criar uma conexão.

Tabela 24. Resumo das interfaces de classe .REDE (continuação)

<b>Interface</b>	<b>Descrição</b>
<a href="#">“IConnectionMetaDados” na página 109</a>	Um objeto de dados ConnectionMeta fornece informações sobre uma conexão.
<a href="#">“IDestination” na página 110</a>	Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.
<a href="#">“ExceptionListener” na página 112</a>	Um aplicativo usa um listener de exceção para ser notificado assincronamente de um problema com uma conexão.
<a href="#">“Exceção IllegalState” na página 112</a>	XMS lança essa exceção se um aplicativo chamar um método em um momento incorreto ou inadequado ou se XMS não estiver em um estado apropriado para a operação solicitada.
<a href="#">“InitialContext” na página 113</a>	Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.
<a href="#">“InvalidClientIDException” na página 115</a>	XMS lança essa exceção se um aplicativo tentar configurar um identificador de cliente para uma conexão, mas o identificador de cliente não for válido ou já estiver em uso.
<a href="#">“Exceção de InvalidDestination” na página 115</a>	XMS lançará essa exceção se um aplicativo especificar um destino que não seja válido
<a href="#">“InvalidSelectorExceção” na página 116</a>	XMS emitirá essa exceção se um aplicativo fornecer uma expressão de seletor de mensagem cuja sintaxe não é válida
<a href="#">“IMapMessage” na página 116</a>	Uma mensagem de mapa é uma mensagem cujo corpo consiste em um conjunto de pares nome-valor, em que cada valor possui um tipo de dados associado.
<a href="#">“IMessage” na página 125</a>	Um objeto de mensagem representa uma mensagem que um aplicativo envia ou recebe. IMessage é uma superclasse para as classes de mensagem, como IMapMessage.
<a href="#">“IMessageConsumer” na página 132</a>	Um aplicativo usa um consumidor de mensagem para receber mensagens enviadas a um destino.
<a href="#">“MessageEOFException” na página 134</a>	XMS lança essa exceção se XMS encontrar o final de um fluxo de mensagens de bytes quando um aplicativo estiver lendo o corpo de uma mensagem de bytes.
<a href="#">“MessageFormatExceção” na página 135</a>	XMS lançará essa exceção se o XMS encontrar uma mensagem com um formato que não seja válido
<a href="#">“IMessageListener (delegado)” na página 135</a>	Um aplicativo usa um listener de mensagens para receber mensagens assincronamente.
<a href="#">“MessageNotReadableException” na página 136</a>	XMS lança essa exceção se um aplicativo tentar ler o corpo de uma mensagem que é somente gravação.

Tabela 24. Resumo das interfaces de classe .REDE (continuação)

Interface	Descrição
<a href="#">“MessageNotWritableException” na página 136</a>	XMS lança essa exceção se um aplicativo tentar gravar no corpo de uma mensagem que é somente leitura.
<a href="#">“IMessageProducer” na página 136</a>	Um aplicativo usa um produtor de mensagem para enviar mensagens para um destino.
<a href="#">“IObjectMessage” na página 142</a>	Uma mensagem de objeto é uma mensagem cujo corpo compreende um objeto Java ou .NET serializado.
<a href="#">“IPropertyContext” na página 143</a>	IPropertyContext é uma superclasse abstrata que contém métodos que obtêm e configuram propriedades. Estes métodos são herdados por outras classes.
<a href="#">“IQueueBrowser” na página 152</a>	Um aplicativo usa um navegador de filas para pesquisar mensagens em uma fila sem removê-las.
<a href="#">“Solicitante” na página 154</a>	Um aplicativo usa um solicitante para enviar uma mensagem de solicitação e, em seguida, aguardar e receber a resposta..
<a href="#">“Exceção de ResourceAllocation” na página 155</a>	XMS lança essa exceção se XMS não puder alocar os recursos necessários por um método.
<a href="#">“SecurityException” na página 156</a>	XMS lança essa exceção se o identificador de usuário e a senha fornecidos para autenticar um aplicativo forem rejeitados XMS também lança essa exceção se uma verificação de autoridade falhar e impedir que um método seja concluído.
<a href="#">“ISession” na página 156</a>	Uma sessão é um único contexto encadeado para enviar e receber mensagens.
<a href="#">“IStreamMessage” na página 167</a>	Uma mensagem de fluxo é uma mensagem cujo corpo compreende um fluxo de valores, em que cada valor possui um tipo de dados associado.
<a href="#">“ITextMessage” na página 176</a>	Uma mensagem de texto é uma mensagem cujo corpo compreende uma sequência.
<a href="#">“TransactionInProgressException” na página 177</a>	XMS lançará essa exceção se um aplicativo solicitar uma operação que não seja válida porque uma transação está em andamento
<a href="#">“TransactionRolledBackException” na página 178</a>	XMS lança esta exceção se um aplicativo chamar Session.commit() para confirmar a transação atual, mas a transação será, então, revertida.
XMSC	Para .REDE, XMS nomes de propriedades e valores são definidos nessa classe como constantes públicas. Para obter detalhes adicionais, consulte <a href="#">“Propriedades de objetos XMS” na página 181</a> .

Tabela 24. Resumo das interfaces de classe .REDE (continuação)

Interface	Descrição
<a href="#">“XMSException” na página 178</a>	<p>Se XMS detectar um erro ao processar uma chamada para um método .REDE , XMS lançará uma exceção. Uma exceção é um objeto que contém informações sobre o erro.</p> <p>Há diferentes tipos de exceção XMS , e um objeto XMSException é apenas um tipo de exceção. Entretanto, a classe XMSException é uma superclasse das outras classes de exceção XMS . XMS lança um objeto XMSException em situações em que nenhum dos outros tipos de exceção é apropriado.</p>
<a href="#">“XMSFactoryFactory” na página 179</a>	Se um aplicativo não estiver usando objetos administrados, utilize essa classe para criar connection factories, filas e tópicos..

A definição de cada método lista os códigos de exceção que o XMS pode retornar se detectar um erro ao processar uma chamada para o método. Cada código de exceções é representado por sua constante nomeada, que possui uma exceção correspondente.

### Conceitos relacionados

[Construindo seus próprios aplicativos](#)

Você constrói seus próprios aplicativos, como você constrói os aplicativos de amostra.

[Gravando aplicativos do XMS](#)

Os tópicos nesta seção fornecem informações para ajudá-lo ao gravar aplicativos XMS ..

[Gravando aplicativos XMS .NET](#)

Os tópicos desta seção fornecem informações para ajudá-lo ao compor aplicativos XMS .NET.

### Referências relacionadas

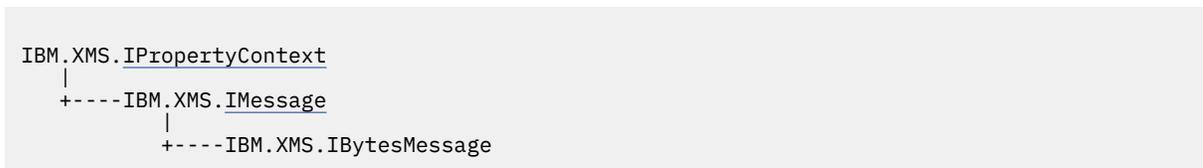
[Propriedades de objetos XMS](#)

Isso capítulo documenta as propriedades de objeto definidas por XMS

## IBytesMessage

Uma mensagem de bytes é uma mensagem cujo corpo compreende um fluxo de bytes.

### Hierarquia de herança:



### Referências relacionadas

[Mensagens de Bytes](#)

O corpo de uma mensagem de bytes contém um fluxo de bytes. O corpo contém apenas os dados reais, e é responsabilidade dos aplicativos de envio e recebimento interpretar esses dados.

### **.REDE propriedades**

### *BodyLength - Obter Comprimento do Corpo*

#### **Interface:**

```
Int64 BodyLength
{
    get;
}
```

Obtenha o comprimento do corpo da mensagem em bytes quando o corpo da mensagem for somente leitura

O valor retornado é o comprimento do corpo inteiro, independentemente de onde o cursor para ler a mensagem está atualmente posicionado.

#### **Exceções:**

- XMSEException
- MessageNotReadableException

### **Methods**

#### *ReadBoolean - Valor Booleano de leitura*

#### **Interface:**

```
Boolean ReadBoolean();
```

Leia um valor booleano do fluxo de mensagens de bytes.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O valor booleano que é lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

#### *ReadSignedByte - Byte de Leitura*

#### **Interface:**

```
Int16 ReadSignedByte();
```

Leia o próximo byte do fluxo de mensagens de bytes como um número inteiro de 8 bits assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O byte lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException

- MessageEOFException

### *ReadBytes -Bytes de leitura*

#### **Interface:**

```
Int32 ReadBytes(Byte[] array);  
Int32 ReadBytes(Byte[] array, Int32 length);
```

Ler uma matriz de bytes a partir do fluxo de mensagens de bytes a partir da posição atual do cursor.

#### **Parâmetros:**

##### **matriz (saída)**

O buffer para conter a matriz de bytes que é lida. Se o número de bytes restantes a serem lidos do fluxo antes da chamada for maior ou igual ao comprimento do buffer, o buffer será preenchido. Caso contrário, o buffer será parcialmente preenchido, com todos os bytes restantes.

Se você especificar um ponteiro nulo na entrada, o método ignorará os bytes sem lê-los. Se o número de bytes restantes a serem lidos a partir do fluxo antes da chamada for maior ou igual ao comprimento do buffer, o número de bytes ignorados será igual ao comprimento do buffer. Caso contrário, todos os bytes restantes serão ignorados.. O cursor permanece na próxima posição para ser lido no fluxo de mensagens de bytes

##### **comprimento (entrada)**

O comprimento do buffer em bytes

#### **Retorna:**

O número de bytes lidos no buffer. Se o buffer estiver parcialmente preenchido, o valor será menor que o comprimento do buffer, indicando que não há mais bytes restantes para serem lidos. Se não houver bytes restantes a serem lidos do fluxo antes da chamada, o valor será XMSC\_END\_OF\_STREAM.

Se você especificar um ponteiro nulo na entrada, o método não retornará valor.

#### **Exceções:**

- XMSEException
- MessageNotReadableException

### *ReadChar -Caractere de Leitura*

#### **Interface:**

```
Char ReadChar();
```

Leia os próximos 2 bytes do fluxo de mensagens de bytes como um caractere.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O caractere lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadDouble -Ler Número de Ponto Flutuante de Precisão Dupla*

#### **Interface:**

```
Double ReadDouble();
```

Leia os próximos 8 bytes do fluxo de mensagens de bytes como um número de ponto flutuante de precisão dupla.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número do ponto flutuante de precisão dupla que é lido

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadFloat -Ler número de ponto flutuante*

#### **Interface:**

```
Single ReadFloat();
```

Leia os próximos 4 bytes do fluxo de mensagens de bytes como um número de ponto flutuante.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número de ponto flutuante que é lido

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadInt -Ler Número Inteiro*

#### **Interface:**

```
Int32 ReadInt();
```

Leia os próximos 4 bytes do fluxo de mensagens de bytes como um número inteiro de 32 bits assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número inteiro que é lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadLong -Número inteiro longo de leitura*

#### **Interface:**

```
Int64 ReadLong();
```

Leia os próximos 8 bytes do fluxo de mensagens de bytes como um número inteiro de 64 bits assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número inteiro longo que é lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadShort -Número Inteiro Curto de Leitura*

#### **Interface:**

```
Int16 ReadShort();
```

Leia os próximos 2 bytes do fluxo de mensagens de bytes como um número inteiro de 16 bits assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número inteiro curto que é lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadByte -Byte não assinado de leitura*

#### **Interface:**

```
Byte ReadByte();
```

Leia o próximo byte do fluxo de mensagens de bytes como um número inteiro de 8 bits não assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O byte lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException

- MessageEOFException

### *ReadUnsignedShort-Read Unsigned Short Integer*

#### **Interface:**

```
Int32 ReadUnsignedShort();
```

Leia os próximos 2 bytes do fluxo de mensagens de bytes como um número inteiro de 16 bits não assinado.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O número inteiro curto não assinado que é lido.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadUTF -Ler Sequência UTF*

#### **Interface:**

```
String ReadUTF();
```

Leia uma cadeia, codificada em UTF-8, a partir do fluxo de mensagens de bytes.

**Nota:** Antes de chamar ReadUTF(), assegure que o cursor do buffer esteja apontando para o início do fluxo de mensagens de byte.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

Um objeto String encapsulando a sequência que é lida.

#### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *Reconfigurar-Reconfigurar*

#### **Interface:**

```
void Reset();
```

Coloque o corpo da mensagem no modo somente leitura e reposicione o cursor no início do fluxo de mensagem de bytes..

#### **Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotReadableException

*WriteBoolean -Gravar Valor Booleano*

**Interface:**

```
void WriteBoolean(Boolean value);
```

Grave um valor booleano no fluxo de mensagens de bytes.

**Parâmetros:****valor (entrada)**

O valor booleano a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteByte -Byte de gravação*

**Interface:**

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

Gravar um byte no fluxo de mensagens de bytes.

**Parâmetros:****valor (entrada)**

O byte a ser gravado..

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteBytes -Bytes de Gravação*

**Interface:**

```
void WriteBytes(Byte[] value);
```

Grave uma matriz de bytes no fluxo de mensagens de bytes.

**Parâmetros:****valor (entrada)**

A matriz de bytes a ser gravada

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteBytes -Matriz de bytes parciais de gravação*

**Interface:**

```
void WriteBytes(Byte[] value, int offset, int length);
```

Grave uma matriz parcial de bytes no fluxo de mensagens de bytes, conforme definido pelo comprimento especificado.

**Parâmetros:****valor (entrada)**

A matriz de bytes a ser gravada

**deslocamento (entrada)**

O ponto de início para a matriz de bytes a ser gravada

**comprimento (entrada)**

O número de bytes a serem gravados

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteChar -Caractere de gravação*

**Interface:**

```
void WriteChar(Char value);
```

Grave um caractere no fluxo de mensagens de bytes como 2 bytes, primeiro byte de alta ordem.

**Parâmetros:****valor (entrada)**

O caractere a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteDouble -Número do ponto flutuante de precisão dupla de gravação*

**Interface:**

```
void WriteDouble(Double value);
```

Converta um número de vírgula flutuante de precisão dupla em um número inteiro longo e grave o número inteiro longo no fluxo de mensagens de bytes como 8 bytes, primeiro byte de alta ordem..

**Parâmetros:**

**valor (entrada)**

O número de ponto flutuante de precisão dupla a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSException
- MessageNotWritableException

*WriteFloat -Número do ponto flutuante de gravação*

**Interface:**

```
void WriteFloat(Single value);
```

Converta um número de vírgula flutuante em um número inteiro e grave o número inteiro no fluxo de mensagens de bytes como 4 bytes, primeiro byte de alta ordem.

**Parâmetros:**

**valor (entrada)**

O número de ponto flutuante a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSException
- MessageNotWritableException

*WriteInt -Número inteiro de gravação*

**Interface:**

```
void WriteInt(Int32 value);
```

Grave um número inteiro no fluxo de mensagens de bytes como 4 bytes, primeiro byte de alta ordem.

**Parâmetros:**

**valor (entrada)**

O número inteiro a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSException

- MessageNotWritableException

*WriteLong -Número inteiro longo de gravação*

**Interface:**

```
void WriteLong(Int64 value);
```

Grave um número inteiro longo no fluxo de mensagens de bytes como 8 bytes, primeiro byte de alta ordem.

**Parâmetros:**

**valor (entrada)**

O número inteiro longo a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteObject -Gravar Objeto*

**Interface:**

```
void WriteObject(Object value);
```

Grave o objeto especificado no fluxo da mensagem de byte

**Parâmetros:**

**valor (entrada)**

O objeto a ser gravado, que deve ser uma referência a um tipo primitivo..

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteShort -Escrever Número Inteiro Curto*

**Interface:**

```
void WriteShort(Int16 value);
```

Grave um número inteiro curto no fluxo de mensagens de bytes como 2 bytes, primeiro byte de alta ordem.

**Parâmetros:**

**valor (entrada)**

O número inteiro curto a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteUTF -Gravar sequência UTF*

**Interface:**

```
void WriteUTF(String value);
```

Grave uma sequência, codificada em UTF-8, para o fluxo de mensagens de bytes

**Parâmetros:****valor (entrada)**

Um objeto String encapsulando a sequência a ser gravada.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

**Propriedades e métodos herdados**

As propriedades a seguir são herdadas da interface IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

Os métodos a seguir são herdados da interface IMessage:

clearBody, clearProperties, PropertyExists

Os métodos a seguir são herdados da interface IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

**ICconnection**

Um objeto Connection representa a conexão ativa do aplicativo para um servidor de sistema de mensagens.

**Hierarquia de herança:**

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.ICconnection
```

Para obter uma lista das propriedades definidas XMS de um objeto Connection, consulte [“Propriedades da Conexão”](#) na página 181.

**.REDE propriedades**

## *ClientID -Obter e Configurar ID do Cliente*

### **Interface:**

```
String ClientID
{
    get;
    set;
}
```

Obter e configurar o identificador de cliente para a conexão.

O identificador de cliente pode ser pré-configurado pelo administrador em um ConnectionFactory ou designado configurando ClientID.

Um identificador de cliente é usado apenas para suportar assinaturas duráveis no domínio de publicação / assinatura e é ignorado no domínio ponto a ponto.

Se um aplicativo configurar um identificador de cliente para uma conexão, o aplicativo deverá fazer isso imediatamente após criar a conexão e antes de executar qualquer outra operação na conexão. Se o aplicativo tentar configurar um identificador de cliente após esse ponto, a chamada emitirá exceção `InvalidOperationException`.

Essa propriedade não é válida para uma conexão em tempo real com um broker

### **Exceções:**

- `XMSEException`
- Exceção `IllegalState`
- `InvalidClientIDException`

## *ExceptionListener -Obter e Configurar Listener de Exceção*

### **Interface:**

```
ExceptionListener ExceptionListener
{
    get;
    set;
}
```

Obtenha o listener de exceção que está registrado com a conexão e registre um listener de exceção com a conexão.

Se nenhum listener de exceção for registrado com a conexão, o método retorna nulo. Se um listener de exceção já estiver registrado com a conexão, será possível cancelar o registro especificando um nulo em vez do listener de exceção.

Para obter mais informações sobre como usar listeners de exceções, consulte [“Listeners de mensagem e de exceção em .NET” na página 49](#)

### **Exceções:**

- `XMSEException`

## *Metadados-Obter metadados*

### **Interface:**

```
IConnectionMetaData MetaData
{
```

```
    get;  
}
```

Obter os metadados da conexão.

**Exceções:**

- XMSEException

## Methods

*Fechar-Fechar Conexão*

**Interface:**

```
void Close();
```

Feche a conexão.

Se um aplicativo tentar fechar uma conexão que já esteja fechada, a chamada será ignorada..

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*CreateSession -Criar Sessão.*

**Interface:**

```
ISession CreateSession(Boolean transacted,  
                        AcknowledgeMode acknowledgeMode);
```

Criar uma sessão

**Parâmetros:**

**transacionado (entrada)**

O valor True significa que a sessão é transacionada O valor False significa que a sessão não foi transacionada.

Para uma conexão em tempo real com um broker, o valor deve ser False.

**acknowledgeMode (entrada).**

Indica como mensagens recebidas por um aplicativo são reconhecidas. O valor deve ser um dos seguintes do enumerador AcknowledgeMode :

```
AcknowledgeMode.AutoAcknowledge  
AcknowledgeMode.ClientAcknowledge  
AcknowledgeMode.DupsOkAcknowledge
```

Para uma conexão em tempo real com um broker, o valor deve ser AcknowledgeMode.AutoAcknowledge ou AcknowledgeMode.DupsOkAcknowledge

Esse parâmetro será ignorado se a sessão for transacionada.. Para obter mais informações sobre os modos de confirmação, consulte [“Reconhecimento de mensagem” na página 26](#)

**Retorna:**

O objeto Session

**Exceções:**

- XMSEException

*Iniciar-Iniciar Conexão***Interface:**

```
void Start();
```

Iniciar ou reiniciar a entrega de mensagens recebidas para a conexão. A chamada será ignorada se a conexão já tiver sido iniciada

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*Parar-Parar Conexão***Interface:**

```
void Stop();
```

Pare a entrega de mensagens recebidas para a conexão. A chamada será ignorada se a conexão já tiver sido interrompida

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**IConnectionFactory**

Um aplicativo usa um connection factory para criar uma conexão.

**Hierarquia de herança:**

```
IBM.XMS.IPropertyContext
```

```
|  
+----IBM.XMS.IConnectionFactory
```

Para obter uma lista das propriedades definidas XMS de um objeto ConnectionFactory , consulte [“Propriedades de ConnectionFactory” na página 182.](#)

### **Conceitos relacionados**

#### ConnectionFactoryies e objetos de Conexão

Um objeto ConnectionFactory fornece um modelo que um aplicativo usa para criar um objeto Connection. O aplicativo usa o objeto Connection para criar um objeto Session.

#### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

#### Conexões seguras com um gerenciador de filas IBM MQ

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um gerenciador de filas IBM MQ , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### Conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um mecanismo do sistema de mensagens WebSphere Application Server service integration bus , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

## **Methods**

*CreateConnection -Criar Connection Factory (utilizando a identidade do usuário padrão).*

### **Interface:**

```
IConnectionFactory CreateConnection();
```

Crie um connection factory com as propriedades padrão..

Se você estiver se conectando ao WebSphere MQ e XMSC\_USERID não estiver configurado, o gerenciador de filas usará o userID do usuário conectado por padrão. Se você precisar de autenticação no nível de conexão adicional de usuários individuais, poderá gravar uma saída de autenticação de cliente que está configurada em WebSphere MQ

### **Parâmetros:**

Nenhum

### **Exceções:**

- XMSException

*CreateConnection - Criar Conexão (usando uma identidade do usuário especificada).*

#### **Interface:**

```
IConnection CreateConnection(String userId, String password);
```

Crie uma conexão usando uma identidade do usuário especificada

Se você estiver se conectando ao WebSphere MQ e XMSC\_USERID não estiver configurado, o gerenciador de filas usará o userID do usuário conectado por padrão. Se você precisar de autenticação no nível de conexão adicional de usuários individuais, poderá gravar uma saída de autenticação de cliente que está configurada em WebSphere MQ

A conexão é criada em modo interrompido Nenhuma mensagem é entregue até que o aplicativo chame **Connection.start()**

#### **Parâmetros:**

##### **userID (entrada)**

Um objeto String que encapsula o identificador de usuário a ser usado para autenticar o aplicativo.. Se você fornecer um valor nulo, será feita uma tentativa de criar a conexão sem autenticação.

##### **senha (entrada)**

Um objeto String encapsulando a senha a ser usada para autenticar o aplicativo. Se você fornecer um valor nulo, será feita uma tentativa de criar a conexão sem autenticação.

#### **Retorna:**

O objeto Conexão.

#### **Exceções:**

- XMSEException
- XMS\_X\_SECURITY\_EXCEPTION

### ***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## **IConnectionMetaDados**

Um objeto de dados ConnectionMeta fornece informações sobre uma conexão.

#### **Hierarquia de herança:**

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionMetaDados
```

Para obter uma lista das propriedades definidas pelo XMS de um objeto de dados ConnectionMeta, consulte [“Propriedades de Dados ConnectionMeta” na página 189](#)

### ***.REDE propriedades***

## Interface:

```
System.Collections.IEnumerator JMSXPropertyNames
{
    get;
}
```

Retornar uma enumeração dos nomes das propriedades de mensagens definidas JMS suportadas pela conexão.

JMS propriedades de mensagem definidas não são suportadas por uma conexão em tempo real com um broker.

## Exceções:

- XMSException

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface `IPropertyContext`:

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IDestination

Um destino é para onde um aplicativo envia mensagens ou é uma origem da qual um aplicativo recebe mensagens, ou ambos.

## Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IDestination
```

Para obter uma lista das propriedades definidas do XMS de um objeto de Destino, consulte [“Propriedades de Destino”](#) na página 189

## Conceitos relacionados

[ConnectionFactory](#)es e objetos de Conexão

Um objeto `ConnectionFactory` fornece um modelo que um aplicativo usa para criar um objeto `Connection`. O aplicativo usa o objeto `Connection` para criar um objeto `Session`.

[Conexão com um barramento de integração de serviços](#)

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

[Destinos](#)

Um aplicativo XMS usa um objeto de Destino para especificar o destino das mensagens que estão sendo enviadas e a origem de mensagens que estão sendo recebidas.

[Curinga de destino](#)

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

[Identificadores de recursos uniformes do tópico](#)

O URI (Identificador Uniforme de Recursos (URI)) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

### Identificadores uniformes de recursos da fila

O URI para uma fila especifica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

### Destinos Temporários

Os aplicativos XMS podem criar e usar destinos temporários.

### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

#### Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

## ***.REDE propriedades***

### *Nome-Obter Nome do Destino*

#### **Interface:**

```
String Name
{
    get;
}
```

Obtenha o nome do destino O nome é uma sequência encapsulando o nome de uma fila ou o nome de um tópico..

#### **Exceções:**

- XMSEException

### *TypeId -Obter Tipo de Destino*

#### **Interface:**

```
DestinationType TypeId
{
    get;
}
```

Obtenha o tipo do destino O tipo do destino é um dos seguintes valores:

DestinationType.Queue  
DestinationType.Topic

#### **Exceções:**

- XMSEException

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## ExceptionHandler

### Hierarquia de herança:

Nenhum

Um aplicativo usa um listener de exceção para ser notificado assincronamente de um problema com uma conexão.

Se um aplicativo usar uma conexão apenas para consumir mensagens de forma assíncrona e sem outro propósito, a única maneira de o aplicativo aprender sobre um problema com a conexão será usando um listener de exceção. Em outras situações, um listener de exceção pode fornecer uma maneira mais imediata de aprender sobre um problema com uma conexão do que esperar até a próxima chamada síncrona para XMS..

## Delegar

*ExceptionHandler -Listener de Exceção*

### Interface:

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

Notificar o aplicativo de um problema com uma conexão.

Os métodos que implementam esse delegado podem ser registrados com a conexão

Para obter mais informações sobre como usar listeners de exceções, consulte [“Listeners de mensagem e de exceção em .NET”](#) na página 49

### Parâmetros:

#### exceção (entrada)

Um ponteiro para uma exceção criada por XMS

### Retorna:

Cancelado

## Exceção IllegalState

### Hierarquia de herança:

```
IBM.XMS.XMSException
|
+----IBM.XMS.Exception
|
+----IBM.XMS.IllegalStateException
```

XMS lança essa exceção se um aplicativo chamar um método em um momento incorreto ou inadequado ou se XMS não estiver em um estado apropriado para a operação solicitada.

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [XMSException](#):

[GetErrorCode](#), [GetLinkedException](#)

## InitialContext

Um aplicativo usa um objeto InitialContext para criar objetos de definições de objeto que são recuperados de um repositório de objetos administrados.

### Hierarquia de herança:

Nenhum

### Conceitos relacionados

#### [Propriedades InitialContext](#)

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### [Formato de URI para contextos iniciais XMS](#)

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### [Recuperação de Objetos Administrados](#)

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

### Tarefas relacionadas

#### [Objetos InitialContext](#)

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

## **.REDE propriedades**

*Ambiente-Obter o ambiente*

### Interface:

```
Hashtable Environment
{
    get;
}
```

Obter o ambiente

### Exceções:

- As exceções são específicas para o serviço de diretório que está sendo usado

## **Construtores**

*InitialContext -Criar Contexto Inicial*

### Interface:

```
InitialContext(Hashtable env);
```

Crie um objeto InitialContext ..

### Parâmetros:

As informações necessárias para estabelecer uma conexão com o repositório de objetos administrados são fornecidas ao construtor em um ambiente Hashtable.

**Exceções:**

- XMSEException

**Methods**

*Ambiente AddTo-Incluir uma nova propriedade no ambiente*

**Interface:**

```
Object AddToEnvironment(String propName, Object propVal);
```

Inclua uma nova propriedade no ambiente

**Parâmetros:****propName (entrada)**

Um objeto String encapsulando o nome da propriedade a ser incluída.

**propVal (entrada)**

O valor da propriedade a ser incluída.

**Retorna:**

O valor antigo da propriedade.

**Exceções:**

- As exceções são específicas para o serviço de diretório que está sendo usado

*Fechar-Fechar este contexto*

**Interface:**

```
void Close()
```

Feche este contexto.

**Parâmetros:**

Nenhum

**Retorna:**

Nenhum

**Exceções:**

- As exceções são específicas para o serviço de diretório que está sendo usado

*Consulta-objeto de consulta no contexto inicial*

**Interface:**

```
Object Lookup(String name);
```

Crie um objeto a partir de uma definição de objeto que é recuperado do repositório de objetos administrados.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome do objeto administrado a ser recuperado. O nome pode ser um nome simples ou um nome complexo. Para obter detalhes adicionais, consulte [“Recuperação de Objetos Administrados” na página 66.](#)

**Retorna:**

Um IConnectionFactory ou um IDestination, dependendo do tipo de objeto sendo recuperado. Se a função puder acessar o diretório, mas não puder localizar o objeto necessário, um nulo será retornado..

**Exceções:**

- As exceções são específicas para o serviço de diretório que está sendo usado

*Ambiente RemoveFrom-Remover uma Propriedade do Ambiente.*

**Interface:**

```
Object RemoveFromEnvironment(String propName);
```

Remove uma propriedade do ambiente.

**Parâmetros:****propName (entrada)**

Um objeto String encapsulando o nome da propriedade a ser removida.

**Retorna:**

O objeto que foi removido..

**Exceções:**

- As exceções são específicas para o serviço de diretório que está sendo usado

## InvalidClientIDException

**Hierarquia de herança:**

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.InvalidClientIDException
```

XMS lança essa exceção se um aplicativo tentar configurar um identificador de cliente para uma conexão, mas o identificador de cliente não for válido ou já estiver em uso.

**Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface [XMSException](#):

[GetErrorCode](#), [GetLinkedException](#)

## Exceção de InvalidDestination

**Hierarquia de herança:**

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.InvalidDestinationException
```

XMS lançará essa exceção se um aplicativo especificar um destino que não seja válido

**Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface [XMSException](#):

[GetErrorCode](#), [GetLinkedException](#)

## InvalidSelectorExceção

### Hierarquia de herança:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
      |
      +----IBM.XMS.InvalidSelectorException
```

XMS emitirá essa exceção se um aplicativo fornecer uma expressão de seletor de mensagem cuja sintaxe não é válida

### Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [XMSException](#):

[GetErrorCode](#), [GetLinkedException](#)

## IMapMessage

Uma mensagem de mapa é uma mensagem cujo corpo consiste em um conjunto de pares nome-valor, em que cada valor possui um tipo de dados associado.

### Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
      |
      +----IBM.XMS.IMapMessage
```

Quando um aplicativo obtém o valor do par nome-valor, o valor pode ser convertido por XMS em outro tipo de dado.. Para obter mais informações sobre essa forma de conversão implícita, consulte [“Mensagens de Mapa” na página 79](#)

### Referências relacionadas

[Mensagens de Mapa](#)

O corpo de uma mensagem de mapa contém um conjunto de pares nome-valor, em que cada valor tem um tipo de dados associado.

### **.REDE** propriedades

*MapNames* -Obter nomes de mapa

#### Interface:

```
System.Collections.IEnumerator MapNames
{
    get;
}
```

Obter uma enumeração dos nomes no corpo da mensagem de mapa.

#### Exceções:

- XMSException

### Methods

*GetBoolean -Obter Valor Booleano*

**Interface:**

```
Boolean GetBoolean(String name);
```

Obtenha o valor booleano identificado pelo nome do corpo da mensagem do mapa.

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome que identifica o valor booleano.

**Retorna:**

O valor booleano recuperado do corpo da mensagem de mapa.

**Exceções:**

- XMSEException

*GetByte -Obter Byte.*

**Interface:**

```
Byte GetByte(String name);  
Int16 GetSignedByte(String name);
```

Obtenha o byte identificado por nome a partir do corpo da mensagem do mapa

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome que identifica o byte.

**Retorna:**

O byte recuperado do corpo da mensagem de mapa. Nenhuma conversão de dados é executada no byte..

**Exceções:**

- XMSEException

*GetBytes -Obter Bytes*

**Interface:**

```
Byte[] GetBytes(String name);
```

Obtenha a matriz de bytes identificados por nome a partir do corpo da mensagem do mapa

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome que identifica a matriz de bytes..

**Retorna:**

O número de bytes na matriz

**Exceções:**

- XMSEException

### *GetChar - Obter Caractere*

#### **Interface:**

```
Char GetChar(String name);
```

Obter o caractere identificado pelo nome a partir do corpo da mensagem do mapa

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome que identifica o caractere.

#### **Retorna:**

O caractere recuperado do corpo da mensagem do mapa.

#### **Exceções:**

- XMSEException

### *GetDouble - Obter Número de Ponto Flutuante de Precisão Dupla*

#### **Interface:**

```
Double GetDouble(String name);
```

Obter o número de vírgula flutuante de precisão dupla identificado por nome a partir do corpo da mensagem do mapa

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome que identifica o número de vírgula flutuante de precisão dupla.

#### **Retorna:**

O número de vírgula flutuante de dupla precisão recuperado do corpo de mensagem do mapa

#### **Exceções:**

- XMSEException

### *GetFloat - Obter número de ponto flutuante*

#### **Interface:**

```
Single GetFloat(String name);
```

Obtenha o número de vírgula flutuante identificado por nome a partir do corpo da mensagem do mapa

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome que identifica o número do ponto flutuante.

#### **Retorna:**

O número de ponto flutuante recuperado do corpo da mensagem do mapa

#### **Exceções:**

- XMSEException

### *GetInt - Obter Número Inteiro*

#### **Interface:**

```
Int32 GetInt(String name);
```

Obter o número inteiro identificado por nome do corpo da mensagem do mapa.

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome que identifica o número inteiro.

#### **Retorna:**

O número inteiro recuperado do corpo da mensagem de mapa.

#### **Exceções:**

- XMSEException

### *GetLong - Obter Número Inteiro Longo*

#### **Interface:**

```
Int64 GetLong(String name);
```

Obter o número inteiro longo identificado pelo nome do corpo da mensagem do mapa.

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome que identifica o número inteiro longo.

#### **Retorna:**

O número inteiro longo recuperado do corpo da mensagem do mapa.

#### **Exceções:**

- XMSEException

### *GetObject - Obter Objeto*

#### **Interface:**

```
Object GetObject(String name);
```

Obter uma referência para o valor de um par nome-valor a partir do corpo da mensagem do mapa. O par de nome-valor é identificado por nome.

#### **Parâmetros:**

##### **nome (entrada)**

Um objeto String encapsulando o nome do par nome-valor.

#### **Retorna:**

O valor, que é um dos seguintes tipos de objeto:

Boolean  
Byte  
Byte[]  
Char  
Double

Single  
Int32  
Int64  
Int16  
String

**Exceções:**

XMSEException

*GetShort -Obter Número Inteiro Curto*

**Interface:**

```
Int16 GetShort(String name);
```

Obter o número inteiro curto identificado por nome a partir do corpo da mensagem do mapa.

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome que identifica o número inteiro curto.

**Retorna:**

O número inteiro curto recuperado do corpo da mensagem do mapa.

**Exceções:**

- XMSEException

*GetString -Obter sequência*

**Interface:**

```
String GetString(String name);
```

Obtenha a sequência identificada pelo nome a partir do corpo da mensagem do mapa

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome que identifica a sequência no corpo da mensagem do mapa.

**Retorna:**

Um objeto String encapsulando a sequência recuperada do corpo de uma mensagem de mapa Se a conversão de dados for necessária, esse valor será a cadeia após a conversão..

**Exceções:**

- XMSEException

*ItemExists -Verifique se o par nome-valor existe*

**Interface:**

```
Boolean ItemExists(String name);
```

Verifique se o corpo da mensagem de mapa contém um par nome-valor com o nome especificado.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome do par nome-valor.

**Retorna:**

- True, se o corpo da mensagem de mapa contiver um par nome-valor com o nome especificado.
- False, se o corpo da mensagem de mapa não contiver um par nome-valor com o nome especificado.

**Exceções:**

- XMSEException

*SetBoolean -Configurar Valor Booleano***Interface:**

```
void SetBoolean(String name, Boolean value);
```

Configure um valor booleano no corpo da mensagem de mapa.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o valor booleano no corpo da mensagem do mapa..

**valor (entrada)**

O valor booleano a ser configurado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetByte -Configurar Byte***Interface:**

```
void SetByte(String name, Byte value);  
void SetSignedByte(String name, Int16 value);
```

Configure um byte no corpo da mensagem do mapa

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o byte no corpo de uma mensagem de mapa

**valor (entrada)**

O byte a ser configurado..

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

## *SetBytes -Configurar Bytes*

### **Interface:**

```
void SetBytes(String name, Byte[] value);
```

Configure uma matriz de bytes no corpo da mensagem de mapa

### **Parâmetros:**

#### **nome (entrada)**

Um objeto String encapsulando o nome para identificar a matriz de bytes no corpo da mensagem do mapa

#### **valor (entrada)**

A matriz de bytes a ser configurada

### **Retorna:**

Cancelado

### **Exceções:**

- XMSEException

## *SetChar -Configurar Caractere*

### **Interface:**

```
void SetChar(String name, Char value);
```

Configure um caractere de 2 bytes no corpo da mensagem de mapeamento.

### **Parâmetros:**

#### **nome (entrada)**

Um objeto String encapsulando o nome para identificar o caractere no corpo da mensagem do mapa

#### **valor (entrada)**

O caractere a ser configurado.

### **Retorna:**

Cancelado

### **Exceções:**

- XMSEException

## *SetDouble -Configurar Número de Ponto Flutuante de Precisão Dupla*

### **Interface:**

```
void SetDouble(String name, Double value);
```

Configure um número de ponto flutuante de precisão dupla no corpo da mensagem do mapa.

### **Parâmetros:**

#### **nome (entrada)**

Um objeto String encapsulando o nome para identificar o número do ponto flutuante de precisão dupla no corpo da mensagem do mapa

#### **valor (entrada)**

O número do ponto flutuante de precisão dupla a ser configurado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetFloat -Configurar número de ponto flutuante***Interface:**

```
void SetFloat(String name, Single value);
```

Configure um número de vírgula flutuante no corpo da mensagem de mapa

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o número de ponto flutuante no corpo da mensagem do mapa

**valor (entrada)**

O número de vírgula flutuante a ser configurado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetInt -Configurar Número Inteiro***Interface:**

```
void SetInt(String name, Int32 value);
```

Configure um número inteiro no corpo da mensagem do mapa.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o número inteiro no corpo da mensagem do mapa.

**valor (entrada)**

O número inteiro a ser configurado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetLong -Configurar número inteiro longo***Interface:**

```
void SetLong(String name, Int64 value);
```

Configure um número inteiro longo no corpo da mensagem de mapa

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o número inteiro longo no corpo da mensagem do mapa..

**valor (entrada)**

O número inteiro longo a ser configurado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetObject -Configurar Objeto***Interface:**

```
void SetObject(String name, Object value);
```

Configure um valor, que deve ser um tipo primitivo XMS , no corpo da mensagem do mapa.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o valor no corpo da mensagem do mapa.

**valor (entrada)**

Uma matriz de bytes contendo o valor a ser configurado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*SetShort -Configurar Número Inteiro Curto***Interface:**

```
void SetShort(String name, Int16 value);
```

Configure um número inteiro curto no corpo da mensagem do mapa.

**Parâmetros:****nome (entrada)**

Um objeto String encapsulando o nome para identificar o número inteiro curto no corpo da mensagem do mapa.

**valor (entrada)**

O número inteiro curto a ser configurado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

## SetString -Configurar Sequência

### Interface:

```
void SetString(String name, String value);
```

Configure uma sequência no corpo da mensagem do mapa.

### Parâmetros:

#### nome (entrada)

Um objeto String encapsulando o nome para identificar a sequência no corpo da mensagem do mapa.

#### valor (entrada)

Um objeto String encapsulando a sequência a ser configurada.

### Retorna:

Cancelado

### Exceções:

- XMSEException

## Propriedades e métodos herdados

As propriedades a seguir são herdadas da interface [IMessage](#):

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedelivered](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Properties](#)

Os métodos a seguir são herdados da interface [IMessage](#):

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IMessage

Um objeto de mensagem representa uma mensagem que um aplicativo envia ou recebe. IMessage é uma superclasse para as classes de mensagem, como [IMapMessage](#).

### Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
```

Para obter uma lista dos campos do cabeçalho da mensagem JMS em um objeto de Mensagem, consulte [“Campos de cabeçalho em uma mensagem XMS”](#) na página 71 Para obter uma lista das propriedades definidas pelo JMS de um objeto de Mensagem, consulte [“JMS-propriedades definidas de uma mensagem”](#) na página 73 Para obter uma lista das propriedades definidas pelo IBM de um objeto de Mensagem, consulte [“IBM-propriedades definidas de uma mensagem”](#) na página 74 Para obter uma lista de propriedades JMS\_IBM\_MQMD\* para o objeto Message, consulte [“Propriedades JMS\\_IBM\\_MQMD\\*”](#) na página 194

As mensagens são excluídas pelo coletor de lixo Quando uma mensagem é excluída, isso libera os recursos que ela estava usando.

## Referências relacionadas

### Mensagens do XMS

Esta seção descreve a estrutura e o conteúdo de mensagens XMS e explica como os aplicativos processam mensagens XMS .

## **.REDE propriedades**

*GetJMSCorrelationID-Obter e Configurar JMSCorrelationID*

### Interface:

```
String JMSCorrelationID
{
    get;
    set;
}
```

Obtenha e configure o identificador de correlação da mensagem como um objeto de Sequência.

### Exceções:

- XMSEException

*JMSDeliveryMode -Obter e configurar JMSDeliveryMode*

### Interface:

```
DeliveryMode JMSDeliveryMode
{
    get;
    set;
}
```

Obter e configurar o modo de entrega da mensagem.

O modo de entrega da mensagem é um dos seguintes valores:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Para uma mensagem recém-criada que não foi enviada, o modo de entrega é `DeliveryMode.Persistente`, exceto para uma conexão em tempo real com um broker para o qual o modo de entrega é `DeliveryMode.NonPersistent` Para uma mensagem recebida, o método retorna o modo de entrega que foi configurado pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere o modo de entrega configurando `JMSDeliveryMode`.

### Exceções:

- XMSEException

*JMSDestination-Obter e Configurar JMSDestination.*

### Interface:

```
IDestination JMSDestination
{
    get;
    set;
}
```

Obtenha e configure o destino da mensagem

O destino é configurado pela chamada de `IMessageProducer.send ()` quando a mensagem é enviada. O valor de `JMSDestination` é ignorado. No entanto, é possível usar `JMSDestination` para alterar o destino de uma mensagem recebida.

Para uma mensagem recém-criada que não foi enviada, o método retorna um objeto `Destino` nulo, a menos que o aplicativo de envio configure um destino configurando `JMSDestination`. Para uma mensagem recebida, o método retorna um objeto `Destination` para o destino que foi configurado pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere o destino configurando `JMSDestination`.

**Exceções:**

- `XMSEException`

*JMSExpiration-Obter e configurar JMSExpiration*

**Interface:**

```
Int64 JMSExpiration
{
    get;
    set;
}
```

Obter e configurar o prazo de expiração da mensagem

O prazo de expiração é configurado pela chamada de `IMessageProducer.send ()` quando a mensagem é enviada. Seu valor é calculado adicionando o tempo de vida, conforme especificado pelo aplicativo de envio, ao tempo em que a mensagem é enviada. O prazo de expiração é expresso em milissegundos desde 00:00:00 GMT de 1 de janeiro de 1970.

Para uma mensagem recém-criada que não foi enviada, o prazo de expiração é 0, a menos que o aplicativo de envio configure um prazo de expiração diferente configurando `JMSExpiration`. Para uma mensagem recebida, o método retorna o prazo de expiração que foi configurado pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere o prazo de expiração configurando `JMSExpiration`.

Se o tempo de vida for 0, a chamada `IMessageProducer.send ()` configurará o tempo de vencimento como 0 para indicar que a mensagem não expira...

O XMS descarta mensagens expiradas e não as entrega para aplicativos.

**Exceções:**

- `XMSEException`

*JMSMessageID -Obter e configurar JMSMessageID*

**Interface:**

```
String JMSMessageID
{
    get;
    set;
}
```

Obtenha e configure o identificador de mensagem da mensagem como um objeto de sequência encapsulando o identificador de mensagens.

O identificador de mensagem é configurado pela chamada `.send ()` de `IMessageProducer` quando a mensagem é enviada. Para uma mensagem que foi recebida, o método retorna o identificador de

mensagem que foi configurado pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere o identificador de mensagem configurando `JMSMessageID...`

Se a mensagem não tiver identificador de mensagem, o método retornará um nulo.

**Exceções:**

- `XMSEException`

*JMSPriority-Obter e configurar JMSPriority.*

**Interface:**

```
Int32 JMSPriority
{
    get;
    set;
}
```

Obtenha e configure a prioridade da mensagem

A prioridade é configurada pela chamada `IMessageProducer.send ()` quando a mensagem é enviada O valor é um número inteiro no intervalo 0, a prioridade mais baixa, para 9, a prioridade mais alta.

Para uma mensagem recém-criada que não foi enviada, a prioridade será 4 a menos que o aplicativo de envio configure uma prioridade diferente configurando `JMSPriority`. Para uma mensagem recebida, o método retorna a prioridade que foi configurada pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere a prioridade configurando `JMSPriority`.

**Exceções:**

- `XMSEException`

*JMSRedelivered-Obter e Configurar JMSRedelivered*

**Interface:**

```
Boolean JMSRedelivered
{
    get;
    set;
}
```

Obter uma indicação se a mensagem está sendo entregue novamente e indicar se a mensagem está sendo entregue novamente. A indicação é configurada pela chamada `IMessageConsumer.receive ()` quando a mensagem é recebida.

Essa propriedade possui os seguintes valores:

- `True`, se a mensagem estiver sendo entregue novamente
- `False`, se a mensagem não estiver sendo entregue novamente

Para uma conexão em tempo real com um broker, o valor é sempre `False`

Uma indicação de nova entrega configurada por `JMSRedelivered` antes de a mensagem ser enviada é ignorada pela chamada `IMessageProducer.send ()` quando a mensagem é enviada e é ignorada e substituída pela chamada `IMessageConsumer.receive ()` quando a mensagem é recebida. No entanto, é possível usar `JMSRedelivered` para alterar a indicação de uma mensagem recebida.

**Exceções:**

- XMSEException

*JMSReplyTo -Get e Configurar JMSReplyTo***Interface:**

```
IDestination JMSReplyTo
{
    get;
    set;
}
```

Obter e configurar o destino para o qual uma resposta à mensagem deve ser enviada.

O valor dessa propriedade é um objeto de Destino para o destino no qual uma resposta à mensagem deve ser enviada. Um objeto Destino nulo significa que nenhuma resposta é esperada.

**Exceções:**

- XMSEException

*JMSTimestamp-Obter e configurar JMSTimestamp.***Interface:**

```
Int64 JMSTimestamp
{
    get;
    set;
}
```

Obter e configurar o horário em que a mensagem foi enviada

O registro de data e hora é configurado pela chamada `IMessageProducer.send ()` quando a mensagem é enviada e é expressa em milissegundos desde 00:00:00 GMT de 1 de janeiro de 1970.

Para uma mensagem recém-criada que não foi enviada, o registro de data e hora será 0, a menos que o aplicativo de envio configure um registro de data e hora diferente configurando `JMSTimestamp`. Para uma mensagem que foi recebida, o método retorna o registro de data e hora que foi configurado pela chamada `IMessageProducer.send ()` quando a mensagem foi enviada, a menos que o aplicativo de recebimento altere o registro de data e hora configurando `JMSTimestamp`.

**Exceções:**

- XMSEException

**Notes:**

1. Se o registro de data e hora for indefinido, o método retornará 0, mas não lançará nenhuma exceção

*JMSType-Obter e Configurar JMSType***Interface:**

```
String JMSType
{
    get;
    set;
}
```

Get e configure o tipo da mensagem.

O valor de JMSType é uma sequência encapsulando o tipo da mensagem... Se a conversão de dados for necessária, esse valor será o tipo após a conversão..

**Exceções:**

- XMSEException

*PropertyNames -Obter Propriedades*

**Interface:**

```
System.Collections.IEnumerator PropertyNames
{
    get;
}
```

Obter uma enumeração das propriedades de nomes da mensagem

**Exceções:**

- XMSEException

**Methods**

*Reconhecimento-Reconhecimento*

**Interface:**

```
void Acknowledge();
```

Reconheça essa mensagem e todas as mensagens não reconhecidas anteriormente recebidas pela sessão.

Um aplicativo pode chamar esse método se o modo de confirmação da sessão for AcknowledgeModeClientAcknowledge. As chamadas para o método serão ignoradas se a sessão tiver qualquer outro modo de reconhecimento ou for transacionada

As mensagens que foram recebidas, mas não reconhecidas, podem ser entregues novamente.

Para obter mais informações sobre o reconhecimento de mensagens, consulte [“Reconhecimento de mensagem”](#) na página 26

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- Exceção IllegalState

### *ClearBody -Limpar corpo*

#### **Interface:**

```
void ClearBody();
```

Limpe o corpo da mensagem. Os campos de cabeçalho e as propriedades de mensagem não são limpos

Se um aplicativo limpar um corpo da mensagem, o corpo permanecerá no mesmo estado que um corpo vazio em uma mensagem recém-criada O estado de um corpo vazio em uma mensagem recém-criada depende do tipo de corpo da mensagem.. Para obter mais informações, consulte [“O corpo de uma mensagem XMS” na página 76](#).

Um aplicativo pode limpar um corpo de mensagem a qualquer momento, não importa em qual estado o corpo está. Se um corpo da mensagem for somente leitura, a única maneira de um aplicativo poder gravar no corpo será para o aplicativo limpar o corpo primeiro.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

Cancelado

#### **Exceções:**

- XMSEException

### *ClearProperties -Limpar Propriedades*

#### **Interface:**

```
void ClearProperties();
```

limpa as propriedades da mensagem. Os campos de cabeçalho e o corpo da mensagem não são limpos

Se um aplicativo limpar as propriedades de uma mensagem, as propriedades se tornarão legíveis e graváveis

Um aplicativo pode limpar as propriedades de uma mensagem a qualquer momento, independentemente do estado em que as propriedades estão. Se as propriedades de uma mensagem forem somente leitura, a única maneira de as propriedades se tornarem graváveis será para o aplicativo limpar as propriedades primeiro.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

Cancelado

#### **Exceções:**

- XMSEException

### *PropertyExists -Verificar propriedade existe*

#### **Interface:**

```
Boolean PropertyExists(String propertyName);
```

Verifique se a mensagem possui uma propriedade com o nome especificado.

## Parâmetros:

### **propertyName (entrada)**

Um objeto String encapsulando o nome da propriedade.

## Retorna:

- True, se a mensagem tiver uma propriedade com o nome especificado
- False, se a mensagem não tiver uma propriedade com o nome especificado

## Exceções:

- XMSEException

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IMessageConsumer

Um aplicativo usa um consumidor de mensagem para receber mensagens enviadas a um destino.

### Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessageConsumer
```

Para obter uma lista das propriedades definidas do XMS de um objeto `MessageConsumer`, consulte [“Propriedades de MessageConsumer”](#) na página 198

## .REDE propriedades

*MessageListener - Obter e Configurar Listener de Mensagens*

### Interface:

```
MessageListener MessageListener
{
    get;
    set;
}
```

Obter o listener de mensagens que está registrado com o consumidor de mensagem e registrar um listener de mensagens com o consumidor de mensagem.

Se nenhum listener de mensagens estiver registrado com o consumidor de mensagens, `MessageListener` será nulo. Se um listener de mensagem já estiver registrado com o consumidor de mensagens, será possível cancelar o registro especificando um nulo.

Para obter mais informações sobre o uso de listeners de mensagens, consulte [“Listeners de mensagem e de exceção em .NET”](#) na página 49

### Exceções:

- XMSEException

## *MessageSelector -Obter Seletor de Mensagem*

### **Interface:**

```
String MessageSelector
{
    get;
}
```

Obter o seletor de mensagem para o consumidor de mensagens O valor de retorno é um objeto String encapsulando a expressão do seletor de mensagem. Se a conversão de dados for necessária, esse valor será a expressão do seletor de mensagem após a conversão.. Se o consumidor de mensagens não tiver um seletor de mensagem, o valor de MessageSelector será um objeto String nulo.

### **Exceções:**

- XMSEException

## **Methods**

*Fechar-Fechar Consumidor da Mensagem.*

### **Interface:**

```
void Close();
```

Feche o consumidor de mensagens.

Se um aplicativo tentar fechar um consumidor de mensagens que já esteja encerrado, a chamada será ignorada

### **Parâmetros:**

Nenhum

### **Retorna:**

Cancelado

### **Exceções:**

- XMSEException

*Receber-Receber*

### **Interface:**

```
IMessage Receive();
```

Receber a próxima mensagem para o consumidor de mensagens A chamada espera indefinidamente por uma mensagem ou até o consumidor de mensagem ser fechado.

### **Parâmetros:**

Nenhum

### **Retorna:**

Um ponteiro para o objeto de Mensagem Se o consumidor de mensagens for fechado enquanto a chamada estiver aguardando uma mensagem, o método retornará um ponteiro para um objeto Mensagem nulo.

### **Exceções:**

- XMSEException

*Recebimento-Recebimento (com um intervalo de espera)*

**Interface:**

```
IMessage Receive(Int64 delay);
```

Receber a próxima mensagem para o consumidor de mensagens A chamada aguarda apenas um período especificado para uma mensagem ou até que o consumidor de mensagens seja fechado.

**Parâmetros:**

**atraso (entrada)**

O tempo, em milissegundos, que a chamada espera por uma mensagem Se você especificar um intervalo de espera de 0, a chamada aguardará indefinidamente uma mensagem.

**Retorna:**

Um ponteiro para o objeto de Mensagem Se nenhuma mensagem chegar durante o intervalo de espera ou se o consumidor de mensagens for fechado enquanto a chamada estiver aguardando uma mensagem, o método retornará um ponteiro para um objeto de Mensagem nulo, mas não emitirá nenhuma exceção.

**Exceções:**

- XMSEException

*ReceiveNoEspera-Receber com Nenhuma Espera*

**Interface:**

```
IMessage ReceiveNoWait();
```

Receba a próxima mensagem para o consumidor de mensagens se uma estiver disponível imediatamente.

**Parâmetros:**

Nenhum

**Retorna:**

Um ponteiro para um objeto de Mensagem Se nenhuma mensagem estiver disponível imediatamente, o método retornará um ponteiro para um objeto Mensagem nulo.

**Exceções:**

- XMSEException

**Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface IPROPERTYContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

**MessageEOFException**

**Hierarquia de herança:**

```
IBM.XMS.XMSEException
```

```

|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.MessageEOFException

```

XMS lança essa exceção se XMS encontrar o final de um fluxo de mensagens de bytes quando um aplicativo estiver lendo o corpo de uma mensagem de bytes.

### ***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## **MessageFormatException**

### **Hierarquia de herança:**

```

IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.MessageFormatException

```

XMS lançará essa exceção se o XMS encontrar uma mensagem com um formato que não seja válido

### ***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## **IMessageListener (delegado)**

### **Hierarquia de herança:**

Nenhum

Um aplicativo usa um listener de mensagens para receber mensagens assincronamente.

### ***Delegar***

*MessageListener -Listener de Mensagens*

### **Interface:**

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

Entregar uma mensagem de forma assíncrona ao consumidor de mensagem.

Os métodos que implementam esse delegado podem ser registrados com a conexão

Para obter mais informações sobre o uso de listeners de mensagens, consulte [“Listeners de mensagem e de exceção em .NET”](#) na página 49

### **Parâmetros:**

#### **mesg (entrada)**

O objeto de Mensagem

### **Retorna:**

Cancelado

## MessageNotReadableException

### Hierarquia de herança:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.MessageNotReadableException
```

XMS lança essa exceção se um aplicativo tentar ler o corpo de uma mensagem que é somente gravação.

### **Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## MessageNotWritableException

### Hierarquia de herança:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.MessageNotWritableException
```

XMS lança essa exceção se um aplicativo tentar gravar no corpo de uma mensagem que é somente leitura.

### **Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## IMessageProducer

Um aplicativo usa um produtor de mensagem para enviar mensagens para um destino.

### Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageProducer
```

Para obter uma lista das propriedades definidas XMS de um objeto MessageProducer , consulte [“Propriedades do MessageProducer”](#) na página 198.

### **.REDE propriedades**

*DeliveryMode -Obter e Configurar Modo de Entrega Padrão*

#### Interface:

```
DeliveryMode DeliveryMode
{
    get;
    set;
}
```

Obter e configurar o modo de entrega padrão para mensagens enviadas pelo produtor de mensagens

O modo de entrega padrão possui um dos seguintes valores:

```
DeliveryMode.Persistent  
DeliveryMode.NonPersistent
```

Para uma conexão em tempo real com um broker, o valor deve ser `DeliveryMode.NonPersistent`.

O valor padrão é `DeliveryMode.Persistent`, exceto para uma conexão em tempo real com um broker para o qual o valor padrão é `DeliveryMode.NonPersistent`.

**Exceções:**

- `XMSEException`

*Destino-Obter Destino*

**Interface:**

```
IDestination Destination  
{  
    get;  
}
```

Obter o destino para o produtor de mensagem

**Parâmetros:**

Nenhum

**Retorna:**

O objeto de Destino Se o produtor da mensagem não tiver um destino, o método retornará um objeto de Destino nulo...

**Exceções:**

- `XMSEException`

*ID de DisableMsg-Obter e Configurar Sinalizador de ID de Mensagem de Desabilitação*

**Interface:**

```
Boolean DisableMessageID  
{  
    get;  
    set;  
}
```

Obter uma indicação de se um aplicativo de recebimento requer que identificadores de mensagens sejam incluídos em mensagens enviadas pelo produtor de mensagens e indicar se um aplicativo de recebimento requer que identificadores de mensagens sejam incluídos em mensagens enviadas pelo produtor de mensagem.

Em uma conexão com um gerenciador de fila ou em uma conexão em tempo real com um broker, esse sinalizador é ignorado. Em uma conexão com um barramento de integração de serviços, o sinalizador é honrado

O ID DisabledMsg possui os valores a seguir:

- `True`, se um aplicativo de recebimento não requerer que identificadores de mensagem sejam incluídos em mensagens enviadas pelo produtor de mensagem
- `False`, se um aplicativo de recebimento exigir que identificadores de mensagem sejam incluídos em mensagens enviadas pelo produtor de mensagens

**Exceções:**

- XMSEException

*DisableMsgTS-Obter e Configurar Sinalizador de Registro de Data e Hora de Desativação*

**Interface:**

```
Boolean DisableMessageTimestamp
{
    get;
    set;
}
```

Obter uma indicação se um aplicativo de recebimento requer que registros de data e hora sejam incluídos em mensagens enviadas pelo produtor de mensagens e indicar se um aplicativo de recebimento requer que registros de data e hora sejam incluídos em mensagens enviadas pelo produtor de mensagem.

Em uma conexão em tempo real com um broker, essa sinalização é ignorada. Em uma conexão com um gerenciador de filas ou em uma conexão com um barramento de integração de serviços, a sinalização é honrada.

DisableMsgTS possui os seguintes valores:

- `True`, se um aplicativo de recebimento não precisar que registros de data e hora sejam incluídos em mensagens enviadas pelo produtor de mensagem
- `False`, se um aplicativo de recebimento não precisar que registros de data e hora sejam incluídos em mensagens enviadas pelo produtor de mensagem

**Retorna:****Exceções:**

- XMSEException

*Prioridade-Obter e Configurar Prioridade Padrão*

**Interface:**

```
Int32 Priority
{
    get;
    set;
}
```

Obter e configurar a prioridade padrão para mensagens enviadas pelo produtor de mensagem

O valor da prioridade da mensagem padrão é um número inteiro no intervalo 0, a prioridade mais baixa, para 9, a prioridade mais alta.

Em uma conexão em tempo real com um broker, a prioridade de uma mensagem é ignorada

**Exceções:**

- XMSEException

*TimeToLive-Obter e configurar o tempo de vida padrão*

**Interface:**

```
Int64 TimeToLive
{
```

```
    get;  
    set;  
}
```

Obter e configurar o período de tempo padrão que uma mensagem existe antes de expirar.

O tempo é medido a partir de quando o produtor da mensagem envia a mensagem e é o tempo padrão de vida em milissegundos. Um valor 0 significa que uma mensagem nunca expira.

Para uma conexão em tempo real com um broker, este valor é sempre 0.

**Exceções:**

- XMSEException

## **Methods**

### *Fechar-Fechar Produtor de Mensagem*

**Interface:**

```
void Close();
```

Feche o produtor da mensagem.

Se um aplicativo tentar fechar um produtor de mensagem que já está fechado, a chamada será ignorada.

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

### *Enviar-Enviar*

**Interface:**

```
void Send(IMessage msg) ;
```

Enviar uma mensagem para o destino especificado quando o produtor da mensagem foi criado. Envie a mensagem usando o modo de entrega padrão, prioridade e tempo de vida do produtor da mensagem.

**Parâmetros:**

**msg (entrada)**

O objeto de Mensagem

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageFormatException
- Exceção de InvalidDestination

*Enviar-Enviar (especificando um modo de entrega, prioridade e tempo de vida)*

**Interface:**

```
void Send(IMessage msg,
         DeliveryMode deliveryMode,
         Int32 priority,
         Int64 timeToLive);
```

Enviar uma mensagem para o destino especificado quando o produtor da mensagem foi criado. Envie a mensagem usando o modo de entrega, prioridade e tempo de vida especificados.

**Parâmetros:**

**msg (entrada)**

O objeto de Mensagem

**deliveryMode (entrada)**

O modo de entrega para a mensagem, que deve ser um dos seguintes valores:

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

Para uma conexão em tempo real com um broker, o valor deve ser DeliveryMode.NonPersistent.

**prioridade (entrada)**

A prioridade da mensagem. O valor pode ser um número inteiro no intervalo 0, para a prioridade mais baixo, para 9, para a prioridade mais alta. Em uma conexão em tempo real com um broker, o valor é ignorado

**timeToAtivo (entrada)**

O tempo de vida da mensagem em milissegundos. Um valor 0 significa que a mensagem nunca expira. Para uma conexão em tempo real com um broker, o valor deve ser 0

**Retorna:**

Cancelado

**Exceções:**

- XMSException
- MessageFormatException
- Exceção de InvalidDestination
- Exceção IllegalState

*Enviar-Enviar (para um destino especificado)*

**Interface:**

```
void Send(IDestination dest, IMessage msg) ;
```

Enviar uma mensagem para um destino especificado se estiver usando um produtor de mensagem para o qual nenhum destino foi especificado quando o produtor de mensagem foi criado. Envie a mensagem usando o modo de entrega padrão, prioridade e tempo de vida do produtor da mensagem.

Geralmente, você especifica um destino ao criar um produtor de mensagens, mas, se não o fizer, deverá especificar um destino toda vez que enviar uma mensagem.

**Parâmetros:**

**dest (entrada)**

O objeto de Destino

**msg (entrada)**

O objeto de Mensagem

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageFormatException
- Exceção de InvalidDestination

*Enviar-Enviar (para um destino especificado, especificando um modo de entrega, prioridade e tempo de vida)*

**Interface:**

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

Enviar uma mensagem para um destino especificado se estiver usando um produtor de mensagem para o qual nenhum destino foi especificado quando o produtor de mensagem foi criado. Envie a mensagem usando o modo de entrega, prioridade e tempo de vida especificados.

Geralmente, você especifica um destino ao criar um produtor de mensagens, mas, se não o fizer, deverá especificar um destino toda vez que enviar uma mensagem.

**Parâmetros:****dest (entrada)**

O objeto de Destino

**msg (entrada)**

O objeto de Mensagem

**deliveryMode (entrada)**

O modo de entrega para a mensagem, que deve ser um dos seguintes valores:

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

Para uma conexão em tempo real com um broker, o valor deve ser DeliveryMode.NonPersistent.

**prioridade (entrada)**

A prioridade da mensagem. O valor pode ser um número inteiro no intervalo 0, para a prioridade mais baixa, para 9, para a prioridade mais alta. Em uma conexão em tempo real com um broker, o valor é ignorado

**timeToAtivo (entrada)**

O tempo de vida da mensagem em milissegundos. Um valor 0 significa que a mensagem nunca expira. Para uma conexão em tempo real com um broker, o valor deve ser 0

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageFormatException
- Exceção de InvalidDestination

- Exceção `IllegalState`

### ***Propriedades e métodos herdados***

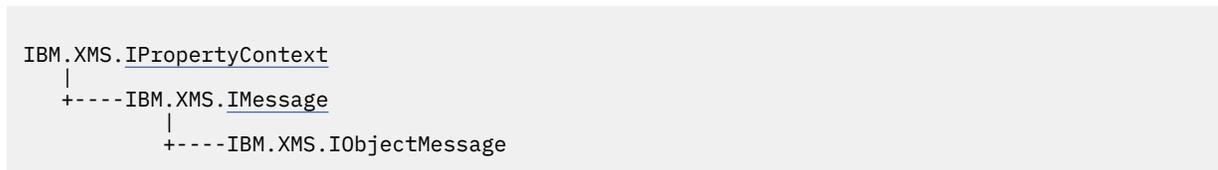
Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## **IObjectMessage**

Uma mensagem de objeto é uma mensagem cujo corpo compreende um objeto Java ou .NET serializado.

### **Hierarquia de herança:**



### **Referências relacionadas**

Mensagens de objeto

O corpo de uma mensagem de objeto contém um objeto serializadoJava ou .NET .

### ***.REDE propriedades***

*Objeto-Obter e Configurar Objeto como Bytes.*

### **Interface:**

```

System.Object Object
{
    get;
    set;
}

Byte[] GetObject();
  
```

Get e configure o objeto que forma o corpo da mensagem do objeto.

### **Exceções:**

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`
- `MessageNotWritableException`

### ***Propriedades e métodos herdados***

As propriedades a seguir são herdadas da interface [IMessage](#):

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSEExpiration](#), [JMSPriority](#), [JMSMessageID](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Properties](#)

Os métodos a seguir são herdados da interface [IMessage](#):

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Os métodos a seguir são herdados da interface [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 é uma superclasse abstrata que contém métodos que obtêm e configuram propriedades. Estes métodos são herdados por outras classes.

### Hierarquia de herança:

Nenhum

### Methods

*Propriedade GetBoolean-Obter Propriedade Booleana*

#### Interface:

```
Boolean GetBooleanProperty(String property_name);
```

Obter o valor da propriedade booleana com o nome especificado

#### Parâmetros:

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

#### Retorna:

O valor da propriedade.

#### Contexto do encadeamento:

Determinado pela subclasse

#### Exceções:

- XMSEException

*Propriedade GetByte-Obter Propriedade de Byte*

#### Interface:

```
Byte    GetByteProperty(String property_name) ;  
Int16   GetSignedByteProperty(String property_name) ;
```

Obter o valor da propriedade de byte identificada por nome.

#### Parâmetros:

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

#### Retorna:

O valor da propriedade.

#### Contexto do encadeamento:

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetBytes-Propriedade da Matriz de Bytes de Obtenção*

**Interface:**

```
Byte[] GetBytesProperty(String property_name) ;
```

Obtenha o valor da propriedade de matriz de bytes identificada por nome..

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O número de bytes na matriz

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetChar-Propriedade de caractere de obtenção*

**Interface:**

```
Char GetCharProperty(String property_name) ;
```

Obter o valor da propriedade de caractere de 2 bytes identificada por nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetDouble-Obter Propriedade de Ponto Flutuante de Precisão Dupla*

**Interface:**

```
Double GetDoubleProperty(String property_name) ;
```

Obtenha o valor da propriedade de ponto flutuante de precisão dupla identificada por nome...

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetFloat-Obter propriedade de ponto flutuante*

**Interface:**

```
Single GetFloatProperty(String property_name) ;
```

Obter o valor da propriedade de ponto flutuante identificada por nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetInt-Propriedade GetInt*

**Interface:**

```
Int32 GetIntProperty(String property_name) ;
```

Obter o valor da propriedade de número inteiro identificada pelo nome

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetLong-Obter propriedade de número inteiro longo*

**Interface:**

```
Int64 GetLongProperty(String property_name) ;
```

Obter o valor da propriedade de número inteiro longo identificado por nome.

**Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetObject-Obter Propriedade do Objeto*

**Interface:**

```
Object GetObjectProperty( String property_name) ;
```

Obtenha o valor e o tipo de dados da propriedade identificados por nome..

**Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade, que é um dos seguintes tipos de objeto:

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetShort-Obter propriedade de número inteiro curto*

**Interface:**

```
Int16 GetShortProperty(String property_name) ;
```

Obter o valor da propriedade de número inteiro curto identificada por nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

O valor da propriedade.

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade GetString-Propriedade GetString*

**Interface:**

```
String GetStringProperty(String property_name) ;
```

Obter o valor da propriedade de sequência identificada por nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**Retorna:**

Um objeto String encapsulando a sequência que é o valor da propriedade. Se a conversão de dados for necessária, esse valor será a cadeia após a conversão..

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException

*Propriedade SetBoolean-Configurar Propriedade Booleana*

**Interface:**

```
void SetBooleanProperty( String property_name, Boolean value) ;
```

Configure o valor da propriedade booleana identificada por nome

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

### *Propriedade SetByte-Configurar propriedade de byte*

#### **Interface:**

```
void SetByteProperty( String property_name, Byte value) ;  
void SetSignedByteProperty( String property_name, Int16 value) ;
```

Configure o valor da propriedade de byte identificada por nome..

#### **Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

#### **Retorna:**

Cancelado

#### **Contexto do encadeamento:**

Determinado pela subclasse

#### **Exceções:**

- XMSEException
- MessageNotWritableException

### *Propriedade SetBytes-Configurar Propriedade da Matriz de Bytes*

#### **Interface:**

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

Configure o valor da propriedade de matriz de bytes identificada por nome..

#### **Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade, que é uma matriz de bytes.

#### **Retorna:**

Cancelado

#### **Contexto do encadeamento:**

Determinado pela subclasse

#### **Exceções:**

- XMSEException
- MessageNotWritableException

### *Propriedade SetChar-Configurar Propriedade de Caractere*

#### **Interface:**

```
void SetCharProperty( String property_name, Char value) ;
```

Configure o valor da propriedade de caractere de 2 bytes identificada por nome..

**Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

*Propriedade SetDouble-Configurar Propriedade de Ponto Flutuante de Precisão Dupla*

**Interface:**

```
void SetDoubleProperty( String property_name, Double value) ;
```

Configure o valor da propriedade de ponto flutuante de precisão dupla identificado por nome..

**Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

*Propriedade SetFloat-Configurar propriedade de ponto flutuante*

**Interface:**

```
void SetFloatProperty( String property_name, Single value) ;
```

Configure o valor da propriedade de ponto flutuante identificada por nome..

**Parâmetros:**

**property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

*Propriedade SetInt-Configurar propriedade de número inteiro*

**Interface:**

```
void SetIntProperty( String property_name, Int32 value) ;
```

Configure o valor da propriedade de número inteiro identificada por nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

*Propriedade SetLong-Configurar propriedade de número inteiro longo*

**Interface:**

```
void SetLongProperty( String property_name, Int64 value) ;
```

Configurar o valor da propriedade de número inteiro longo identificada pelo nome.

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

### *Propriedade SetObject-Configurar Propriedade do objeto*

#### **Interface:**

```
void SetObjectProperty( String property_name, Object value) ;
```

Configure o valor e o tipo de dado de uma propriedade identificada por nome.

#### **Parâmetros:**

##### **property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

##### **objectType (entrada)**

O valor da propriedade, que deve ser um dos seguintes tipos de objeto:

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

##### **valor (entrada)**

O valor da propriedade como uma matriz de bytes.

##### **comprimento (entrada)**

O número de bytes na matriz

#### **Retorna:**

Cancelado

#### **Contexto do encadeamento:**

Determinado pela subclasse

#### **Exceções:**

- XMSEException
- MessageNotWritableException

### *Propriedade SetShort-Configurar propriedade de número inteiro curto*

#### **Interface:**

```
void SetShortProperty( String property_name, Int16 value) ;
```

Configure o valor da propriedade de número inteiro curto identificada por nome..

#### **Parâmetros:**

##### **property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

##### **valor (entrada)**

O valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

*Propriedade SetString-Configurar Propriedade de Cadeia*

**Interface:**

```
void SetStringProperty( String property_name, String value);
```

Configure o valor da propriedade de cadeia identificada por nome..

**Parâmetros:****property\_name (entrada)**

Um objeto String encapsulando o nome da propriedade.

**valor (entrada)**

Um objeto String encapsulando a sequência que é o valor da propriedade.

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Determinado pela subclasse

**Exceções:**

- XMSEException
- MessageNotWritableException

## IQueueBrowser

Um aplicativo usa um navegador de filas para pesquisar mensagens em uma fila sem removê-las.

**Hierarquia de herança:**

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
|
+---- IBM.XMS.IQueueBrowser
```

## Propriedades .REDE

*MessageSelector -Obter Seletor de Mensagem*

**Interface:**

```
String MessageSelector
{
    get;
}
```

Obtenha o seletor de mensagem do navegador de fila.

O seletor de mensagem é um objeto String que encapsula a expressão do seletor de mensagem. Se a conversão de dados for necessária, esse valor será a expressão do seletor de mensagem após a

conversão.. Se o navegador da fila não tiver um seletor de mensagens, o método retornará um objeto String nulo.

**Exceções:**

- XMSEException

*Fila-Fila de obtenção*

**Interface:**

```
IDestination Queue
{
    get;
}
```

Obtenha a fila associada ao navegador de fila como um objeto de destino que representa a fila

**Exceções:**

- XMSEException

**Methods**

*Fechar-Fechar Navegador da Fila*

**Interface:**

```
void Close();
```

Feche o navegador da filas.

Se um aplicativo tentar fechar um browser de fila que já esteja fechado, a chamada será ignorada

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*GetEnumerator -Obter mensagens*

**Interface:**

```
IEnumerator GetEnumerator();
```

Obter uma lista das mensagens na fila

O método retorna um enumerador que encapsula uma lista de objetos de Mensagem. A ordem dos objetos de Mensagem é igual à ordem na qual as mensagens seriam recuperadas da fila. O aplicativo pode então usar o enumerador para pesquisar cada mensagem por vez.

O enumerador é atualizado dinamicamente conforme as mensagens são colocadas na fila e removidas da fila. Toda vez que o aplicativo chama `IEnumerator.MoveNext ()` para procurar a próxima mensagem na fila, a mensagem reflete o conteúdo atual da fila.

Se um aplicativo chamar esse método mais de uma vez para um navegador de filas, cada chamada retornará um novo enumerador. O aplicativo pode, portanto, usar mais de um enumerador para procurar as mensagens em uma fila e manter várias posições dentro da fila.

**Parâmetros:**

Nenhum

**Retorna:**

O objeto do Iterator (Iterator)

**Exceções:**

- XMSEException

### ***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

### **Solicitante**

Um aplicativo usa um solicitante para enviar uma mensagem de solicitação e, em seguida, aguardar e receber a resposta..

**Hierarquia de herança:**

Nenhum

### ***Construtores***

*Solicitante-Criar Solicitante*

**Interface:**

```
Requestor(ISession sess, IDestination dest);
```

Criar um solicitante.

**Parâmetros:**

**sess (entrada)**

Um objeto de Sessão. A sessão não deve ser transacionada e deve ter um dos seguintes modos de confirmação:

- AcknowledgeMode.AutoAcknowledge
- AcknowledgeMode.DupsOkAcknowledge

**dest (entrada)**

Um objeto de Destino representando o destino no qual o aplicativo pode enviar mensagens de solicitação.

**Contexto do encadeamento:**

A sessão associada ao solicitante

**Exceções:**

- XMSEException

### ***Methods***

## Fechar-Fechar Solicitante

### Interface:

```
void Close();
```

Feche o solicitante.

Se um aplicativo tentar fechar um solicitante que já esteja fechado, a chamada será ignorada..

**Nota:** Quando um aplicativo fecha um solicitante, a sessão associada não é fechada também Nesse aspecto, XMS se comporta de forma diferente em comparação com JMS.

### Parâmetros:

Nenhum

### Retorna:

Cancelado

### Contexto do encadeamento:

Qualquer

### Exceções:

- XMSEException

## Solicitação-Resposta de solicitação

### Interface:

```
IMessage Request(IMessage requestMessage);
```

Envie uma mensagem de solicitação e, em seguida, aguarde e receba uma resposta do aplicativo que recebe a mensagem de solicitação

Uma chamada para esse método é bloqueada até que uma resposta seja recebida ou até que a sessão termine, o que ocorrer primeiro.

### Parâmetros:

#### **requestMessage (entrada)**

O objeto de Mensagem encapsulando a mensagem de solicitação

### Retorna:

Um ponteiro para o objeto de Mensagem que contém a mensagem de resposta.

### Contexto do encadeamento:

A sessão associada ao solicitante

### Exceções:

- XMSEException

## Exceção de ResourceAllocation

### Hierarquia de herança:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.ResourceAllocationException
```

XMS lança essa exceção se XMS não puder alocar os recursos necessários por um método.

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## SecurityException

### Hierarquia de herança:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.SecurityException
```

O XMS lança essa exceção se o identificador de usuário e a senha fornecidos para autenticar um aplicativo forem rejeitados XMS também lança essa exceção se uma verificação de autoridade falhar e impedir que um método seja concluído.

## Propriedades e métodos herdados

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## ISession

Uma sessão é um único contexto encadeado para enviar e receber mensagens.

### Hierarquia de herança:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.ISession
```

Para obter uma lista das propriedades definidas XMS de um objeto Session, consulte [“Propriedades da Sessão.”](#) na página 198.

## Propriedades .NET

*AcknowledgeMode - Obter Modo de Confirmação*

### Interface:

```
AcknowledgeMode AcknowledgeMode
{
    get;
}
```

Obter o modo de confirmação da sessão.

O modo de confirmação é especificado quando a sessão é criada

Desde que a sessão não seja transacionada, o modo de reconhecimento será um dos seguintes valores:

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

Para obter mais informações sobre os modos de confirmação, consulte [“Reconhecimento de mensagem”](#) na página 26

Uma sessão transacionada não possui modo de confirmação. Se a sessão for transacionada, o método retornará `AcknowledgeMode.SessionTransacted`.

**Exceções:**

- `XMSEException`

*Transacionado-Determine se Transacionado*

**Interface:**

```
Boolean Transacted
{
    get;
}
```

Determine se a sessão foi transacionada

O transacionado declarado é:

- `True`, se a sessão for transacionada..
- `False`, se a sessão não for transacionada..

Para uma conexão em tempo real com um broker, o método sempre retorna `False`..

**Exceções:**

- `XMSEException`

**Methods**

*Fechar-Fechar Sessão*

**Interface:**

```
void Close();
```

Feche a sessão. Se a sessão for transacionada, qualquer transação em andamento será retrocedida

Se um aplicativo tentar fechar uma sessão que já tenha sido fechada, a chamada será ignorada

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Contexto do encadeamento:**

Qualquer

**Exceções:**

- `XMSEException`

*Confirmação-Confirmar*

**Interface:**

```
void Commit();
```

Confirmar todas as mensagens processada na transação atual

A sessão deve ser uma sessão transacionada

**Parâmetros:**

Nenhum

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- Exceção IllegalStateException
- TransactionRolledBackException

*CreateBrowser -Criar Navegador da Fila*

**Interface:**

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

Crie um navegador de filas da fila especificada.

**Parâmetros:**

**fila (entrada)**

Um objeto de Destino que representa a fila

**Retorna:**

O objeto QueueBrowser .

**Exceções:**

- XMSEException
- Exceção de InvalidDestination

*CreateBrowser -Create Queue Browser (com seletor de mensagens)*

**Interface:**

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

Crie um navegador de fila para a fila especificada usando um seletor de mensagem

**Parâmetros:**

**fila (entrada)**

Um objeto de Destino que representa a fila

**seletor (entrada)**

Um objeto String encapsulando uma expressão do seletor de mensagem. Apenas as mensagens com propriedades que correspondem à expressão do seletor de mensagem são entregues para o navegador de filas

Um objeto String nulo significa que não há nenhum seletor de mensagens para o navegador de fila

**Retorna:**

O objeto QueueBrowser .

**Exceções:**

- XMSEException
- Exceção de InvalidDestination

- InvalidSelectorExceção

### *Mensagem CreateBytes-Criar Mensagem de Bytes*

#### **Interface:**

```
IBytesMessage CreateBytesMessage();
```

Crie uma mensagem de bytes

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto BytesMessage ..

#### **Exceções:**

- XMSException
- IllegalStateExceção (A sessão foi encerrada)

### *CreateConsumer -Criar Consumidor*

#### **Interface:**

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

Crie um consumidor de mensagens para o destino especificado

#### **Parâmetros:**

##### **dest (entrada)**

O objeto de Destino

#### **Retorna:**

O objeto MessageConsumer ..

#### **Exceções:**

- XMSException
- Exceção de InvalidDestination

### *CreateConsumer -Criar Consumidor (com seletor de mensagens).*

#### **Interface:**

```
IMessageConsumer CreateConsumer(IDestination dest,  
String selector) ;
```

Criar um consumidor de mensagens para o destino especificado usando um seletor de mensagens

#### **Parâmetros:**

##### **dest (entrada)**

O objeto de Destino

##### **seletor (entrada)**

Um objeto String encapsulando uma expressão do seletor de mensagem. Apenas as mensagens com propriedades que correspondem à expressão do seletor de mensagem são entregues para o consumidor de mensagem

Um objeto String nulo significa que não há seletor de mensagem para o consumidor de mensagens.

**Retorna:**

O objeto MessageConsumer ..

**Exceções:**

- XMSEException
- Exceção de InvalidDestination
- InvalidSelectorExceção

*CreateConsumer -Criar Consumidor (com seletor de mensagens e sinalização de mensagem local)*

**Interface:**

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector,
                                Boolean noLocal) ;
```

Crie um consumidor de mensagens para o destino especificado utilizando um seletor de mensagens e, se o destino for um tópico, especificando se o consumidor de mensagens recebe as mensagens publicadas pela sua própria conexão

**Parâmetros:**

**dest (entrada)**

O objeto de Destino

**selector (entrada)**

Um objeto String encapsulando uma expressão do seletor de mensagem. Apenas as mensagens com propriedades que correspondem à expressão do seletor de mensagem são entregues para o consumidor de mensagem

Um objeto String nulo significa que não há seletor de mensagem para o consumidor de mensagens.

**noLocal (entrada)**

O valor True significa que o consumidor de mensagens não recebe as mensagens publicadas por sua própria conexão O valor False significa que o consumidor de mensagens recebe as mensagens publicadas por sua conexão. O valor padrão é False.

**Retorna:**

O objeto MessageConsumer ..

**Exceções:**

- XMSEException
- Exceção de InvalidDestination
- InvalidSelectorExceção

*CreateDurableAssinante-Criar Assinante Durável*

**Interface:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription) ;
```

Crie um assinante durável para o tópico especificado

Este método não é válido para uma conexão em tempo real com um broker

Para obter mais informações sobre assinantes duráveis, consulte [“Assinantes duráveis” na página 34..](#)

**Parâmetros:**

**dest (entrada)**

Um objeto de Destino que representa o tópico O tópico não deve ser temporário.

**assinatura (entrada)**

Um objeto String encapsulando um nome que identifica a assinatura durável. O nome deve ser exclusivo no identificador de cliente para a conexão.

**Retorna:**

O objeto MessageConsumer representando o assinante durável.

**Exceções:**

- XMSEException
- Exceção de InvalidDestination

*CreateDurableAssinante-Criar Assinante Durável (com seletor de mensagem e sinalizador de mensagem local)*

**Interface:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                         String subscription,  
                                         String selector,  
                                         Boolean noLocal) ;
```

Crie um assinante durável para o tópico especificado usando um seletor de mensagem e especificando se o assinante durável recebe as mensagens publicadas pela sua própria conexão

Este método não é válido para uma conexão em tempo real com um broker

Para obter mais informações sobre assinantes duráveis, consulte [“Assinantes duráveis” na página 34..](#)

**Parâmetros:**

**dest (entrada)**

Um objeto de Destino que representa o tópico O tópico não deve ser temporário.

**assinatura (entrada)**

Um objeto String encapsulando um nome que identifica a assinatura durável. O nome deve ser exclusivo no identificador de cliente para a conexão.

**seletor (entrada)**

Um objeto String encapsulando uma expressão do seletor de mensagem. Apenas as mensagens com propriedades que correspondem à expressão do seletor de mensagem são entregues para o assinante durável

Um objeto String nulo significa que não há seletor de mensagem para o assinante durável.

**noLocal (entrada)**

O valor True significa que o assinante durável não recebe as mensagens publicadas por sua própria conexão O valor False significa que o assinante durável recebe as mensagens publicadas por sua conexão. O valor padrão é False.

**Retorna:**

O objeto MessageConsumer representando o assinante durável.

**Exceções:**

- XMSEException
- Exceção de InvalidDestination
- InvalidSelectorExceção

### *Mensagem CreateMap-Criar Mensagem de Mapa*

#### **Interface:**

```
IMapMessage CreateMapMessage();
```

Crie uma mensagem de mapa.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto MapMessage ..

#### **Exceções:**

- XMSEException
- IllegalStateException (A sessão foi encerrada)

### *CreateMessage -Criar mensagem*

#### **Interface:**

```
IMessage CreateMessage();
```

Crie uma mensagem que não tenha corpo.

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto de Mensagem

#### **Exceções:**

- XMSEException
- IllegalStateException (A sessão foi encerrada)

### *Mensagem CreateObject-Criar Mensagem de Objeto*

#### **Interface:**

```
IObjectMessage CreateObjectMessage();
```

Criar uma mensagem de objeto

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto ObjectMessage ..

#### **Exceções:**

- XMSEException
- IllegalStateException (A sessão foi encerrada)

### *CreateProducer -Criar Produtor*

#### **Interface:**

```
IMessageProducer CreateProducer(IDestination dest) ;
```

Crie um produtor de mensagem para enviar mensagens ao destino especificado.

#### **Parâmetros:**

##### **dest (entrada)**

O objeto de Destino

Se você especificar um objeto de Destino nulo, o produtor de mensagem será criado sem um destino Nesse caso, o aplicativo deve especificar um destino sempre que usar o produtor de mensagem para enviar uma mensagem.

#### **Retorna:**

O objeto MessageProducer .

#### **Exceções:**

- XMSException
- Exceção de InvalidDestination

### *CreateQueue -Criar Fila*

#### **Interface:**

```
IDestination CreateQueue(String queue) ;
```

Crie um objeto de Destino para representar uma fila no servidor de sistema de mensagens

Este método não cria a fila no servidor de mensagens. Deve-se criar a fila antes que um aplicativo possa chamar esse método

#### **Parâmetros:**

##### **fila (entrada)**

Um objeto String encapsulando o nome da fila ou encapsulando um identificador uniforme de recursos (URI) que identifica a fila.

#### **Retorna:**

O objeto de Destino que representa a fila

#### **Exceções:**

- XMSException

### *Mensagem CreateStream-Criar Mensagem de Fluxo*

#### **Interface:**

```
IStreamMessage CreateStreamMessage();
```

Criar uma mensagem de fluxo

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto StreamMessage ..

**Exceções:**

- XMSEException
- XMS\_ILLEGAL\_STATE\_EXCEPTION

*Fila CreateTemporary-Criar Fila Temporária***Interface:**

```
IDestination CreateTemporaryQueue() ;
```

Crie uma fila temporária.

O escopo da fila temporária é a conexão. Apenas as sessões criadas pela conexão podem usar a fila temporária.

A fila temporária permanece até que seja explicitamente excluída ou a conexão termine, o que for anterior.

Para obter mais informações sobre as filas temporárias, consulte [“Destinos Temporários” na página 32](#)

**Parâmetros:**

Nenhum

**Retorna:**

O objeto de Destino que representa a fila temporária

**Exceções:**

- XMSEException

*Tópico CreateTemporary-Criar tópico temporário***Interface:**

```
IDestination CreateTemporaryTopic() ;
```

Crie um tópico temporário

O escopo do tópico temporário é a conexão. Apenas as sessões criadas pela conexão podem usar o tópico temporário.

O tópico temporário permanece até que seja excluído explicitamente ou a conexão seja encerrada, o que ocorrer primeiro.

Para obter mais informações sobre tópicos temporários, consulte [“Destinos Temporários” na página 32..](#)

**Parâmetros:**

Nenhum

**Retorna:**

O objeto de Destino que representa o tópico temporário

**Exceções:**

- XMSEException

### *Mensagem CreateText-Criar Mensagem de Texto*

#### **Interface:**

```
ITextMessage CreateTextMessage();
```

Crie uma mensagem de texto com um corpo vazio

#### **Parâmetros:**

Nenhum

#### **Retorna:**

O objeto TextMessage ..

#### **Exceções:**

- XMSEException

### *Mensagem CreateText-Criar Mensagem de Texto (inicializada)*

#### **Interface:**

```
ITextMessage CreateTextMessage(String initialValue);
```

Crie uma mensagem de texto cujo corpo seja inicializado com o texto especificado

#### **Parâmetros:**

##### **initialValue (entrada)**

Um objeto String encapsulando o texto para inicializar o corpo da mensagem de texto.

Nenhum

#### **Retorna:**

O objeto TextMessage ..

#### **Exceções:**

- XMSEException

### *CreateTopic -Criar Tópico*

#### **Interface:**

```
IDestination CreateTopic(String topic) ;
```

Crie um objeto de Destino para representar um tópico

#### **Parâmetros:**

##### **tópico (entrada)**

Um objeto String encapsulando o nome do tópico ou encapsulando um identificador uniforme de recursos (URI) que identifica o tópico.

#### **Retorna:**

O objeto de Destino que representa o tópico

#### **Exceções:**

- XMSEException

## *Recuperar-Recuperar*

### **Interface:**

```
void Recover();
```

Recuperar a sessão. A entrega de mensagem é interrompida e, em seguida, reiniciada com a mensagem não reconhecida mais antiga

A sessão não deve ser uma sessão transacionada

Para obter mais informações sobre a recuperação de uma sessão, consulte [“Reconhecimento de mensagem”](#) na página 26

### **Parâmetros:**

Nenhum

### **Retorna:**

Cancelado

### **Exceções:**

- XMSEException
- Exceção IllegalState

## *Retrocesso-Retrocesso*

### **Interface:**

```
void Rollback();
```

Retorne todas as mensagens processadas na transação atual.

A sessão deve ser uma sessão transacionada

### **Parâmetros:**

Nenhum

### **Retorna:**

Cancelado

### **Exceções:**

- XMSEException
- Exceção IllegalState

## *Cancelar assinatura-Cancelar assinatura*

### **Interface:**

```
void Unsubscribe(String subscription);
```

Excluir uma assinatura durável. O servidor de mensagens exclui o registro da assinatura durável que ele está mantendo e não envia mais mensagens para o assinante durável.

Um aplicativo não pode excluir uma assinatura durável em qualquer uma das seguintes circunstâncias:

- Enquanto há um consumidor de mensagens ativo para a assinatura durável
- Enquanto uma mensagem consumida faz parte de uma transação pendente
- Enquanto uma mensagem consumida não foi reconhecida

Este método não é válido para uma conexão em tempo real com um broker

**Parâmetros:**

**assinatura (entrada)**

Um objeto String encapsulando o nome que identifica a assinatura durável.

**Retorna:**

Cancelado

**Exceções:**

- XMSException
- Exceção de InvalidDestination
- Exceção IllegalState

**Propriedades e métodos herdados**

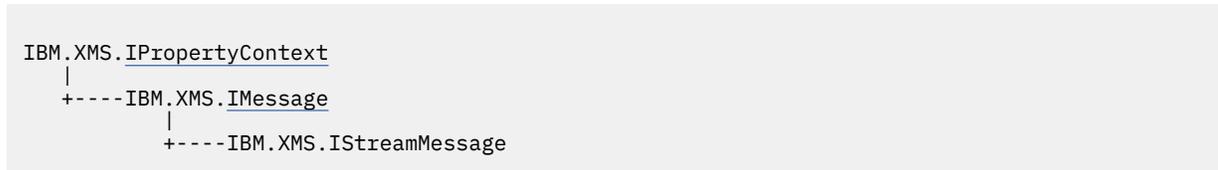
Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**IStreamMessage**

Uma mensagem de fluxo é uma mensagem cujo corpo compreende um fluxo de valores, em que cada valor possui um tipo de dados associado. Os conteúdos do corpo são gravados e lidos sequencialmente.

**Hierarquia de herança:**



Quando um aplicativo lê um valor do fluxo de mensagens, o valor pode ser convertido pelo XMS em outro tipo de dados. Para obter mais informações sobre essa forma de conversão implícita, consulte [“Mensagens de Fluxo” na página 80](#)

**Referências relacionadas**

[Mensagens de Fluxo](#)

O corpo de uma mensagem de fluxo contém um fluxo de valores, em que cada valor possui um tipo de dados associado.

**Methods**

*ReadBoolean -Valor Booleano de leitura*

**Interface:**

```
Boolean ReadBoolean();
```

Leia um valor booleano do fluxo de mensagens.

**Parâmetros:**

Nenhum

**Retorna:**

O valor booleano que é lido.

**Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadByte -Byte de Leitura*

**Interface:**

```
Int16  ReadSignedByte();  
Byte   ReadByte();
```

Leia um número inteiro de 8 bits assinado do fluxo de mensagens.

**Parâmetros:**

Nenhum

**Retorna:**

O byte lido.

**Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadBytes -Bytes de leitura*

**Interface:**

```
Int32  ReadBytes(Byte[] array);
```

Ler uma matriz de bytes do fluxo de mensagens.

**Parâmetros:****matriz (entrada)**

O buffer contendo a matriz de bytes que é lida e o comprimento do buffer em bytes.

Se o número de bytes na matriz for menor ou igual ao comprimento do buffer, a matriz inteira será lida no buffer. Se o número de bytes na matriz for maior que o comprimento do buffer, o buffer será preenchido com parte da matriz e um cursor interno marcará a posição do byte seguinte a ser lido. Uma chamada subsequente para readBytes() lê bytes a partir da matriz iniciando na posição atual do cursor.

Se você especificar um ponteiro nulo na entrada, a chamada ignorará a matriz de bytes sem lê-la.

**Retorna:**

O número de bytes lidos no buffer. Se o buffer for parcialmente preenchido, o valor será menor que o comprimento do buffer, indicando que não há mais bytes na matriz restantes a serem lidos. Se não houver bytes restantes a serem lidos na matriz antes da chamada, o valor será XMSC\_END\_OF\_BYTEARRAY.

Se você especificar um ponteiro nulo na entrada, o método não retornará valor.

**Exceções:**

- XMSEException
- MessageNotReadableException

- MessageEOFException

#### *ReadChar -Caractere de Leitura*

##### **Interface:**

```
Char ReadChar();
```

Leia um caractere de 2 bytes do fluxo de mensagens.

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O caractere lido.

##### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

#### *ReadDouble -Ler Número de Ponto Flutuante de Precisão Dupla*

##### **Interface:**

```
Double ReadDouble();
```

Leia um número de ponto flutuante de precisão dupla de 8 bytes a partir do fluxo de mensagens

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O número do ponto flutuante de precisão dupla que é lido

##### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

#### *ReadFloat -Ler número de ponto flutuante*

##### **Interface:**

```
Single ReadFloat();
```

Leia um número de ponto flutuante de 4 bytes a partir do fluxo de mensagens

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O número de ponto flutuante que é lido

##### **Exceções:**

- XMSEException

- MessageNotReadableException
- MessageEOFException

#### *ReadInt -Ler Número Inteiro*

##### **Interface:**

```
Int32 ReadInt();
```

Leia um número inteiro assinado de 32 bits do fluxo de mensagens.

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O número inteiro que é lido.

##### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

#### *ReadLong -Número inteiro longo de leitura*

##### **Interface:**

```
Int64 ReadLong();
```

Leia um número inteiro assinado de 64 bits a partir do fluxo de mensagens.

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O número inteiro longo que é lido.

##### **Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

#### *ReadObject -Ler Objeto*

##### **Interface:**

```
Object ReadObject();
```

Leia um valor do fluxo de mensagens e retorne seu tipo de dados.

##### **Parâmetros:**

Nenhum

##### **Retorna:**

O valor, que é um dos seguintes tipos de objeto:

Boolean

Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**Exceções:**

XMSEException

*ReadShort -Número Inteiro Curto de Leitura*

**Interface:**

```
Int16 ReadShort();
```

Leia um número inteiro de 16 bits assinado a partir do fluxo de mensagens.

**Parâmetros:**

Nenhum

**Retorna:**

O número inteiro curto que é lido.

**Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadString -Sequência de Leitura*

**Interface:**

```
String ReadString();
```

Leia uma cadeia do fluxo de mensagens. Se necessário, XMS converte os caracteres na sequência na página de código local.

**Parâmetros:**

Nenhum

**Retorna:**

Um objeto String encapsulando a sequência que é lida. Se a conversão de dados for necessária, esta será a sequência após a conversão

**Exceções:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

## Reconfigurar-Reconfigurar

### Interface:

```
void Reset();
```

Coloque o corpo da mensagem no modo somente leitura e posicione o cursor no início do fluxo de mensagens.

### Parâmetros:

Nenhum

### Retorna:

Cancelado

### Exceções:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## WriteBoolean -Gravar Valor Booleano

### Interface:

```
void WriteBoolean(Boolean value);
```

Grave um valor booleano no fluxo de mensagens.

### Parâmetros:

#### valor (entrada)

O valor booleano a ser gravado.

### Retorna:

Cancelado

### Exceções:

- XMSEException
- MessageNotWritableException

## WriteByte -Byte de gravação

### Interface:

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

Grave um byte no fluxo de mensagens

### Parâmetros:

#### valor (entrada)

O byte a ser gravado..

### Retorna:

Cancelado

### Exceções:

- XMSEException

- MessageNotWritableException

*WriteBytes -Bytes de Gravação*

**Interface:**

```
void WriteBytes(Byte[] value);
```

Gravar uma matriz de bytes no fluxo de mensagens.

**Parâmetros:**

**valor (entrada)**

A matriz de bytes a ser gravada

**comprimento (entrada)**

O número de bytes na matriz

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteChar -Caractere de gravação*

**Interface:**

```
void WriteChar(Char value);
```

Grave um caractere no fluxo de mensagens como 2 bytes, primeiro byte de alta ordem.

**Parâmetros:**

**valor (entrada)**

O caractere a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteDouble -Número do ponto flutuante de precisão dupla de gravação*

**Interface:**

```
void WriteDouble(Double value);
```

Converta um número de vírgula flutuante de precisão dupla em um número inteiro longo e grave o número inteiro longo no fluxo de mensagem como 8 bytes, primeiro byte de alta ordem...

**Parâmetros:**

**valor (entrada)**

O número de ponto flutuante de precisão dupla a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteFloat -Número do ponto flutuante de gravação*

**Interface:**

```
void WriteFloat(Single value);
```

Converta um número de ponto flutuante em um número inteiro e escreva o número inteiro no fluxo de mensagens como 4 bytes, primeiro byte de alta ordem.

**Parâmetros:****valor (entrada)**

O número de ponto flutuante a ser gravado

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteInt -Número inteiro de gravação*

**Interface:**

```
void WriteInt(Int32 value);
```

Escreva um número inteiro no fluxo de mensagem como 4 bytes, primeiro byte de alta ordem.

**Parâmetros:****valor (entrada)**

O número inteiro a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteLong -Número inteiro longo de gravação*

**Interface:**

```
void WriteLong(Int64 value);
```

Grave um número inteiro longo no fluxo de mensagens como 8 bytes, primeiro byte de alta ordem.

**Parâmetros:****valor (entrada)**

O número inteiro longo a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteObject -Gravar Objeto*

**Interface:**

```
void WriteObject(Object value);
```

Grave um valor, com um tipo de dado especificado, no fluxo de mensagens.

**Parâmetros:****objectType (entrada)**

O valor, que deve ser um dos seguintes tipos de objeto:

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**valor (entrada)**

Uma matriz de bytes contendo o valor a ser gravado.

**comprimento (entrada)**

O número de bytes na matriz

**Retorna:**

Cancelado

**Exceções:**

- XMSEException

*WriteShort -Escrever Número Inteiro Curto*

**Interface:**

```
void WriteShort(Int16 value);
```

Grave um número inteiro curto no fluxo de mensagens como 2 bytes, primeiro byte de alta ordem.

**Parâmetros:****valor (entrada)**

O número inteiro curto a ser gravado.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

*WriteString - Sequência de Gravação*

**Interface:**

```
void WriteString(String value);
```

Grave uma sequência no fluxo de mensagens.

**Parâmetros:****valor (entrada)**

Um objeto String encapsulando a sequência a ser gravada.

**Retorna:**

Cancelado

**Exceções:**

- XMSEException
- MessageNotWritableException

**Propriedades e métodos herdados**

As propriedades a seguir são herdadas da interface [IMessage](#):

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedelivered](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Properties](#)

Os métodos a seguir são herdados da interface [IMessage](#):

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**ITextMessage**

Uma mensagem de texto é uma mensagem cujo corpo compreende uma sequência.

**Hierarquia de herança:**

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
      |
      +---- IBM.XMS.ITextMessage
```

## Referências relacionadas

Mensagens de texto

O corpo de uma mensagem de texto contém uma sequência.

## **.REDE propriedades**

*Texto-Obter e Configurar Texto*

### Interface:

```
String Text
{
    get;
    set;
}
```

Get e configure a sequência que forma o corpo da mensagem de texto.

Se necessário, XMS converte os caracteres na sequência na página de código local.

### Exceções:

- [XMSEException](#)
- [MessageNotReadableException](#)
- [MessageNotWritableException](#)
- [MessageEOFException](#)

## **Propriedades e métodos herdados**

As propriedades a seguir são herdadas da interface [IMessage](#):

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedelivered](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Properties](#)

Os métodos a seguir são herdados da interface [IMessage](#):

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Os métodos a seguir são herdados da interface [IPropertyContext](#):

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## **TransactionInProgressException**

### Hierarquia de herança:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.TransactionInProgressException
```

XMS lançará essa exceção se um aplicativo solicitar uma operação que não seja válida porque uma transação está em andamento

## **Propriedades e métodos herdados**

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## TransactionRolledBackException

**Hierarquia de herança:**

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.TransactionRolledBackException
```

XMS lança esta exceção se um aplicativo chamar `Session.commit()` para confirmar a transação atual, mas a transação será, então, revertida.

### ***Propriedades e métodos herdados***

Os métodos a seguir são herdados da interface [XMSEException](#):

[GetErrorCode](#), [GetLinkedException](#)

## XMSEException

Se XMS detectar um erro ao processar uma chamada para um método `.REDE`, XMS lançará uma exceção. Uma exceção é um objeto que contém informações sobre o erro.

**Hierarquia de herança:**

```
System.Exception
|
+----IBM.XMS.XMSEException
```

Há diferentes tipos de exceção XMS, e um objeto `XMSEException` é apenas um tipo de exceção. Entretanto, a classe `XMSEException` é uma superclasse das outras classes de exceção XMS. XMS lança um objeto `XMSEException` em situações em que nenhum dos outros tipos de exceção é apropriado.

### ***.REDE propriedades***

*ErrorCode* - Obter Código de Erro

**Interface:**

```
public String ErrorCode
{
    get {return errorCode_;}
}
```

Obter o código de erro

**Exceções:**

- `XMSEException`

*LinkedException* - Obter Exceção Vinculada

**Interface:**

```
public Exception LinkedException
{
    get { return linkedException_;}
```

```
set { linkedException_ = value; }  
}
```

Obter a próxima exceção na cadeia de exceção.

O método retornará um nulo se não houver mais exceções na cadeia.

**Exceções:**

- XMSEException

## **XMSFactoryFactory**

Se um aplicativo não estiver usando objetos administrados, utilize essa classe para criar connection factories, filas e tópicos..

**Hierarquia de herança:**

Nenhum

### ***.REDE propriedades***

*Metadados-Recuperar metadados*

**Interface:**

```
IConnectionMetaData MetaData
```

Obtenha os metadados apropriados para o tipo de conexão do objeto XMSFactoryFactory .

**Exceções:**

Nenhum

### ***Methods***

*CreateConnectionFactory-Criar Connection Factory*

**Interface:**

```
IConnectionFactory CreateConnectionFactory();
```

Crie um objeto ConnectionFactory do tipo declarado.

**Parâmetros:**

Nenhum

**Retorna:**

O objeto ConnectionFactory .

**Exceções:**

- XMSEException

*CreateQueue -Criar Fila*

**Interface:**

```
IDestination CreateQueue(String name);
```

Crie um objeto de Destino para representar uma fila no servidor de sistema de mensagens

Este método não cria a fila no servidor de mensagens. Deve-se criar a fila antes que um aplicativo possa chamar esse método

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome da fila ou encapsulando um identificador uniforme de recursos (URI) que identifica a fila.

**Retorna:**

O objeto de Destino que representa a fila

**Exceções:**

- XMSEException

*CreateTopic -Criar Tópico*

**Interface:**

```
IDestination CreateTopic(String name);
```

Crie um objeto de Destino para representar um tópico

**Parâmetros:**

**nome (entrada)**

Um objeto String encapsulando o nome do tópico ou encapsulando um identificador uniforme de recursos (URI) que identifica o tópico.

**Retorna:**

O objeto de Destino que representa o tópico

**Exceções:**

- XMSEException

*GetInstance -Obtenha uma instância de XMSFactoryFactory*

**Interface:**

```
static XMSFactoryFactory GetInstance(int connectionType);
```

Criar uma instância de XMSFactoryFactory. Um aplicativo XMS usa um objeto XMSFactoryFactory para obter uma referência a um objeto ConnectionFactory apropriado para o tipo de protocolo necessário. Esse objeto ConnectionFactory pode então produzir conexões somente para esse tipo de protocolo.

**Parâmetros:**

**connectionType (entrada)**

O tipo de conexão para o qual o objeto ConnectionFactory produz conexões:

- XMSC.CT\_WPM
- XMSC.CT\_RTT
- XMSC.CT\_WMQ

**Retorna:**

O objeto XMSFactoryFactory dedicado ao tipo de conexão declarado.

**Exceções:**

- Exceção NotSupportedException

## Propriedades de objetos XMS

Isso capítulo documenta as propriedades de objeto definidas por XMS

O capítulo contém o seguinte seções:

- [“Propriedades da Conexão” na página 181](#)
- [“Propriedades de ConnectionFactory” na página 182](#)
- [“Propriedades de Dados ConnectionMeta” na página 189](#)
- [“Propriedades de Destino” na página 189](#)
- [“Propriedades de InitialContext” na página 192](#)
- [“Propriedades de Mensagem” na página 192](#)
- [“Propriedades de MessageConsumer” na página 198](#)
- [“Propriedades do MessageProducer” na página 198](#)
- [“Propriedades da Sessão.” na página 198](#)

Cada seção lista as propriedades de um objeto do tipo especificado e fornece uma descrição simples de cada propriedade..

Esse capítulo também contém o [“Definições de propriedades” na página 198](#) seção, que fornece uma definição de cada propriedade.

Se um aplicativo definir suas próprias propriedades dos objetos descritos neste capítulo, ele não causará um erro, mas poderá causar resultados imprevisíveis

**Nota:** Os nomes e valores da propriedade nesta seção são mostrados no formulário XMSC . *NAME*, que é o formulário usado para C e C + +. No entanto, no .NET, o formulário do nome da propriedade pode ser XMSC . *NAME* ou XMSC\_ *NAME*, dependendo de como você está usando:

- Se você estiver especificando uma propriedade, o nome da propriedade deverá estar no formato XMSC . *NAME* conforme mostrado no exemplo a seguir:

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- Se você estiver especificando uma cadeia, o nome da propriedade deverá estar no formato XMSC\_ *NAME*, conforme mostrado no exemplo a seguir:

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

No .NET, os nomes e valores de propriedades são fornecidos como constantes na classe XMSC. Essas constantes identificam sequências e seriam usadas por qualquer aplicativo do XMS .NET. Se você estiver usando essas constantes predefinidas, os nomes e valores de propriedades estarão no formato XMSC . *NAME*, portanto, por exemplo, você usaria XMSC . *USERID*, em vez de XMSC\_ *USERID*.

Os tipos de dados também estão no formato usado para C/C + +. É possível localizar os valores correspondentes para .NET em [“Tipos de dados para .NET” na página 45](#)

### Conceitos relacionados

Construindo seus próprios aplicativos

Você constrói seus próprios aplicativos, como você constrói os aplicativos de amostra.

### Referências relacionadas

[.REDE interfaces](#)

Este seção documenta as interface de classe .REDE e suas propriedades e métodos.

## Propriedades da Conexão

Uma visão geral das propriedades do objeto de Conexão, com links para informações de referência mais detalhadas

<i>Tabela 25. Propriedades da Conexão</i>	
<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">“XMSC_WMQ_RESOLVED_QUEUE_MANAGER” na página 233</a>	Essa propriedade é usada para obter o nome do gerenciador de filas ao qual ele está conectado
<a href="#">“XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID” na página 233</a>	Essa propriedade é preenchida com o ID do Gerenciador de Filas após a conexão.
<a href="#">XMSC_WPM_CONNECTION_PROTOCOL</a>	O protocolo de comunicações usado para a conexão com o mecanismo do sistema de mensagens. Essa propriedade é somente leitura.
<a href="#">XMSC_WPM_HOST_NAME</a>	O nome do host ou o endereço IP do sistema que contém o mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.
<a href="#">XMSC_WPM_ME_NAME</a>	O nome do mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.
<a href="#">XMSC_WPM_PORT</a>	O número da porta atendida pelo mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.

Um objeto Connection também possui propriedades somente leitura que são derivadas das propriedades do connection factory que foi usado para criar a conexão. Essas propriedades são derivadas não apenas das propriedades do connection factory que foram definidas no momento em que a conexão foi criada, mas também dos valores padrão das propriedades não configuradas. As propriedades incluem apenas aquelas relevantes para o tipo de servidor de sistema de mensagens ao qual o aplicativo está conectado. Os nomes das propriedades são iguais aos nomes das propriedades do connection factory.

## Propriedades de ConnectionFactory

Uma visão geral das propriedades do objeto ConnectionFactory, com links para informações de referência mais detalhadas.

<i>Tabela 26. Propriedades de ConnectionFactory</i>	
<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">“XMSC_ASYNC_EXCEPÇÕES” na página 209</a>	Essa propriedade determina se o XMS informa um ExceptionListener apenas quando uma conexão é quebrada ou quando qualquer exceção ocorre de forma assíncrona para uma chamada da API do XMS. Essa propriedade se aplica a todas as Conexões criadas por meio dessa ConnectionFactory que possui um ExceptionListener registrado.
<a href="#">XMSC_CLIENT_ID</a>	O identificador do cliente para uma conexão.
<a href="#">XMSC_CONNECTION_TYPE</a>	O tipo de servidor de mensagens para o qual um aplicativo se conecta.
<a href="#">XMSC_PASSWORD</a>	Uma senha que pode ser usada para autenticar o aplicativo quando ele tenta se conectar a um servidor de mensagens.
<a href="#">“XMSC_RTT_BROKER_PING_INTERVAL” na página 214</a>	O intervalo de tempo, em milissegundos, após o qual o XMS .NET verifica a conexão com um servidor do sistema de mensagens Real Time para detectar qualquer atividade.

<i>Tabela 26. Propriedades de ConnectionFactory (continuação)</i>	
<b>Nome da propriedade</b>	<b>Descrição</b>
<u>XMSC_RTT_CONNECTION_PROTOCOL</u>	O protocolo de comunicações usado para uma conexão em tempo real com um broker.
<u>XMSC_RTT_HOST_NAME</u>	O nome do host ou o endereço IP do sistema no qual um broker é executado.
<u>XMSC_RTT_LOCAL_ADDRESS</u>	O nome do host ou o endereço IP da interface de rede local a ser usada para uma conexão em tempo real com um broker.
<u>XMSC_RTT_MULTICAST</u>	A configuração multicast para um connection factory ou destino.
<u>XMSC_RTT_PORT</u>	O número da porta na qual um broker atende às solicitações recebidas.
<u>XMSC_USERID</u>	Um identificador de usuário que pode ser usado para autenticar o aplicativo quando ele tenta se conectar a um servidor de mensagens.
<u>XMSC_WMQ_BROKER_CONTROLQ</u>	O nome da fila de controle usada por um broker.  <b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7
<u>XMSC_WMQ_BROKER_PUBQ</u>	O nome da fila monitorada por um broker na qual os aplicativos enviam mensagens que publicam.  <b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7
<u>XMSC_WMQ_BROKER_QMGR</u>	O nome do gerenciador de filas ao qual um broker está conectado.  <b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7

Tabela 26. Propriedades de ConnectionFactory (continuação)

Nome da propriedade	Descrição
<a href="#">XMSC_WMQ_BROKER_SUBQ</a>	<p>O nome da fila de assinantes para um consumidor de mensagens não durável.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<a href="#">XMSC_WMQ_BROKER_VERSION</a>	<p>O tipo de corretor usado pelo aplicativo para uma conexão ou para o destino.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<a href="#">"XMSC_WMQ_CCDTURL" na página 219</a>	<p>Um Localizador Uniforme de Recursos (URL) que identifica o nome e o local do arquivo que contém a tabela de definição de canal do cliente e também especifica como o arquivo pode ser acessado.</p>
<a href="#">XMSC_WMQ_CHANEXO L</a>	<p>O nome do canal a ser usado para uma conexão.</p>
<a href="#">"XMSC_WMQ_CLIENT_RECONNECT_OPTIONS" na página 220</a>	<p>Esta propriedade especifica as opções de reconexão do cliente para novas conexões criadas por este factory..</p>
<a href="#">"XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT" na página 220</a>	<p>Esta propriedade especifica a duração de tempo, em segundos, que uma conexão do cliente tenta reconectar.</p>
<a href="#">XMSC_WMQ_CONNECTION_MODE</a>	<p>O modo pelo qual um aplicativo se conecta a um gerenciador de filas.</p>
<a href="#">"XMSC_WMQ_CONNECTION_NAME_LIST" na página 221</a>	<p>Esta propriedade especifica os hosts aos quais o cliente tenta se reconectar depois que sua conexão é interrompida</p>
<a href="#">XMSC_WMQ_FAIL_IF QUIESCE</a>	<p>Se as chamadas para determinados métodos falharão se o gerenciador de filas ao qual o aplicativo está conectado estiver em um estado quiesce.</p>
<a href="#">XMSC_WMQ_HOST_NAME</a>	<p>O nome do host ou o endereço IP do sistema no qual um gerenciador de filas é executado.</p>
<a href="#">XMSC_WMQ_LOCAL_ADDRESS</a>	<p>Para uma conexão com um gerenciador de filas, essa propriedade especifica a interface de rede local a ser usada, a porta local ou o intervalo de portas locais a serem usadas ou ambos.</p>

Tabela 26. Propriedades de ConnectionFactory (continuação)

Nome da propriedade	Descrição
<a href="#">XMSC_WMQ_MESSAGE_SELECTION</a>	<p>Determina se a seleção de mensagem é feita pelo cliente XMS ou pelo broker</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<a href="#">XMSC_WMQ_MSG_BATCH_SIZE</a>	<p>O número máximo de mensagens a serem recuperadas de uma fila em um lote ao usar a entrega de mensagem assíncrona.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<a href="#">XMSC_WMQ_POLLING_INTERVAL</a>	<p>Se cada listener de mensagem dentro de uma sessão não tiver mensagens adequadas em sua fila, este valor será o intervalo máximo, em milissegundos, que decorrerá antes que cada listener da mensagem tente novamente obter uma mensagem de sua fila.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<a href="#">"XMSC_WMQ_PROVIDER_VERSION" na página 230</a>	<p>A versão, liberação, nível de modificação e fix pack do gerenciador de filas ao qual o aplicativo pretende se conectar.</p>
<a href="#">XMSC_WMQ_PORT</a>	<p>O número da porta na qual um gerenciador de filas atende às solicitações recebidas.</p>
<a href="#">XMSC_WMQ_PUB_ACK_INTERVAL</a>	<p>O número de mensagens publicadas por um publicador antes do cliente XMS solicitar uma confirmação do broker.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>

<i>Tabela 26. Propriedades de ConnectionFactory (continuação)</i>	
<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">“XMSC_WMQ_PUT_ASYNC_ALLOWED” na página 226</a>	Essa propriedade determina se os produtores de mensagens têm permissão para usar as postagens assíncronas para enviar mensagens para esse destino.
<a href="#">XMSC_WMQ_QMGR_CCSD</a>	O identificador (CCSID) do conjunto de caracteres codificados, ou página de código, no qual os campos de dados de caracteres definidos na Message Queue Interface (MQI) são trocados entre o cliente XMS e o cliente WebSphere MQ .
<a href="#">XMSC_WMQ_QUEUE_MANAGER</a>	O nome do gerenciador de filas para conexão.
<a href="#">XMSC_WMQ_RECEIVE_EXIT</a>	Identifica uma saída de recebimento do canal para ser executada.
<a href="#">XMSC_WMQ_RECEIVE_EXIT_INIT</a>	Os dados do usuário que são transmitidos para uma saída de recebimento de canal quando ela é chamada.
<a href="#">XMSC_WMQ_SECURITY_EXIT</a>	Identifica uma saída de segurança do canal.
<a href="#">XMSC_WMQ_SECURITY_EXIT_INIT</a>	Os dados do usuário que são transmitidos para uma saída de segurança do canal quando ela é chamada.
<a href="#">“XMSC_WMQ_SEND_CHECK_COUNT” na página 235</a>	O número de chamadas de envio a serem permitidas entre a verificação de erros de postagem assíncrona, dentro de uma única sessão XMS não transacionada.
<a href="#">XMSC_WMQ_SEND_EXIT</a>	Identifica uma saída de envio de canal.
<a href="#">XMSC_WMQ_SEND_EXIT_INIT</a>	Os dados do usuário que são transmitidos para as saídas de envio do canal quando são chamadas.
<a href="#">“XMSC_WMQ_SHARE_CONV_ALLOWED” na página 235</a>	Se uma conexão do cliente pode compartilhar seu soquete com outras conexões XMS de nível superior do mesmo processo para o mesmo gerenciador de filas, se as definições de canais corresponderem Essa propriedade é fornecida para permitir o isolamento completo de Conexões em soquetes separados, se necessário para desenvolvimento de aplicativos, manutenção ou razões operacionais.
<a href="#">XMSC_WMQ_SSL_CERT_STORES</a>	Os locais dos servidores que retêm as listas de revogação de certificado (CRLs) a serem usadas em uma conexão SSL com um gerenciador de filas.
<a href="#">XMSC_WMQ_SSL_CIPHER_SPEC</a>	O nome da CipherSpec a ser usada em uma conexão segura com um gerenciador de filas.
<a href="#">XMSC_WMQ_SSL_CIPHER_SUITE</a>	O nome do CipherSuite a ser usado em uma conexão TLS para um gerenciador de filas. O protocolo usado para negociar a conexão segura depende do CipherSuite especificado.
<a href="#">XMSC_WMQ_SSL_CRYPT_HW</a>	Detalhes de configuração para o hardware criptográfico conectado ao sistema do cliente.

Tabela 26. Propriedades de ConnectionFactory (continuação)

Nome da propriedade	Descrição
<a href="#">XMSC_WMQ_SSL_FIPS_REQUIRED</a>	O valor dessa propriedade determina se um aplicativo pode ou não usar conjuntos de cifras compatíveis não FIPS. Se essa propriedade for configurada como true, apenas algoritmos do FIPS serão usados para a conexão cliente-servidor.
<a href="#">XMSC_WMQ_SSL_KEY_REPOSITORY</a>	O local do arquivo do banco de dados de chaves no qual chaves e certificados são armazenados.
<a href="#">XMSC_WMQ_SSL_KEY_RESETCOUNT</a>	O KeyResetCount representa o número total de bytes não criptografados enviados e recebidos dentro de uma conversa SSL antes de a chave secreta ser renegociada.
<a href="#">XMSC_WMQ_SSL_PEER_NAME</a>	O nome do peer a ser usado em uma conexão SSL com um gerenciador de filas.
<a href="#">XMSC_WMQ_SYNCPOINT_ALL_GETS</a>	Se todas as mensagens devem ser recuperadas de filas dentro do controle de ponto de sincronização.
"XMSC_WMQ_TARGET_CLIENT" na página 243	
<a href="#">XMSC_WMQ_TEMP_Q_PREFIX</a>	O prefixo usado para formar o nome da WebSphere MQ fila dinâmica que é criada quando o aplicativo cria um XMS fila temporária.
<a href="#">XMSC_WMQ_TEMP_TOPIC_PREFIX</a>	Ao criar tópicos temporários, o XMS gera uma sequência de tópicos no formato "TEMP/TEMPTOPICPREFIX/unique_id" ou se essa propriedade contiver o valor padrão, essa sequência, "TEMP/unique_id", será gerada. Especificar um valor não vazio permite que as filas modelo específicas sejam definidas para criar as filas gerenciadas para assinantes de tópicos temporários criados sob essa conexão.
<a href="#">XMSC_WMQ_TEMPORARY_MODEL</a>	O nome da fila modelo do WebSphere MQ a partir da qual uma fila dinâmica é criada quando o aplicativo cria um XMS fila temporária.
<a href="#">XMSC_WPM_BUS_NAME</a>	Para um connection factory, o nome do barramento de integração de serviços ao qual o aplicativo se conecta ou, para um destino, o nome do barramento de integração de serviços no qual o destino existe.
<a href="#">XMSC_WPM_CONNECTION_PROXIMIDADE</a>	A configuração de proximidade de conexão para a conexão.
<a href="#">XMSC_WPM_DUR_SUB_HOME</a>	O nome do mecanismo do sistema de mensagens no qual todas as assinaturas duráveis para uma conexão ou um destino são gerenciadas.
<a href="#">XMSC_WPM_LOCAL_ADDRESS</a>	Para uma conexão com um barramento de integração de serviços, essa propriedade especifica a interface de rede local a ser usada, a porta local ou o intervalo de portas locais a serem usados ou ambos.
<a href="#">XMSC_WPM_NON_PERSISTENT_MAP</a>	O nível de confiabilidade de mensagens não persistentes que são enviadas usando a conexão.
<a href="#">XMSC_WPM_PERSISTENT_MAP</a>	O nível de confiabilidade de mensagens persistentes que são enviadas usando a conexão.

Tabela 26. Propriedades de ConnectionFactory (continuação)

Nome da propriedade	Descrição
<u>XMSC_WPM_PROVIDER_ENDPOINTS</u>	Uma sequência de um ou mais endereços de terminal de servidores de autoinicialização.
<u>XMSC_WPM_TARGET_GROUP</u>	O nome de um grupo de destinos de mecanismos do sistema de mensagens.
<u>XMSC_WPM_TARGET_SIGNIFICATIVO</u>	O significado do grupo de destinos dos mecanismos do sistema de mensagens.
<u>XMSC_WPM_TARGET_TRANSPORT_CHAIN</u>	O nome da cadeia de transporte de entrada que o aplicativo deve usar para se conectar a um mecanismo do sistema de mensagens.
<u>XMSC_WPM_TARGET_TYPE</u>	O tipo do grupo de destinos de mecanismos do sistema de mensagens.
<u>XMSC_WPM_TEMP_Q_PREFIX</u>	O prefixo usado para formar o nome da fila temporária que é criada no barramento de integração de serviços quando o aplicativo cria um XMS fila provisória.
<u>XMSC_WPM_TEMP_TOPIC_PREFIX</u>	O prefixo usado para formar o nome de um tópico temporário que é criado pelo aplicativo.

### Conceitos relacionados

#### ConnectionFactories e objetos de Conexão

Um objeto ConnectionFactory fornece um modelo que um aplicativo usa para criar um objeto Connection. O aplicativo usa o objeto Connection para criar um objeto Session.

#### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

#### Conexões seguras com um gerenciador de filas IBM MQ

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um gerenciador de filas IBM MQ , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### Conexões seguras com um mecanismo de sistema de mensagens WebSphere Application Server service integration bus

Para ativar um aplicativo XMS.NET para fazer conexões seguras com um mecanismo do sistema de mensagens WebSphere Application Server service integration bus , as propriedades relevantes devem ser definidas no objeto ConnectionFactory.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### Tarefas relacionadas

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### Referências relacionadas

#### Propriedades necessárias para objetos ConnectionFactory administrados

Quando um aplicativo cria um connection factory, um número de propriedades deve ser definido para criar uma conexão com um servidor de sistema de mensagens.

## Propriedades de Dados ConnectionMeta

Uma visão geral do objeto de dados ConnectionMeta, com links para informações de referência mais detalhadas.

<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">XMSC_JMS_MAJOR_VERSION</a>	O número da versão principal da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.
<a href="#">XMSC_JMS_MINOR_VERSION</a>	O número da versão secundária da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.
<a href="#">XMSC_JMS_VERSION</a>	O identificador de versão da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.
<a href="#">XMSC_MAJOR_VERSION</a>	O número da versão do cliente do XMS Essa propriedade é somente leitura.
<a href="#">XMSC_MINOR_VERSION</a>	O número da liberação do cliente do XMS Essa propriedade é somente leitura.
<a href="#">XMSC_PROVIDER_NAME</a>	O provedor do cliente XMS . Essa propriedade é somente leitura.
<a href="#">XMSC_VERSION</a>	O identificador de versão do cliente XMS . Essa propriedade é somente leitura.

## Propriedades de Destino

Uma visão geral das propriedades do objeto de Destino, com links para informações de referência mais detalhadas

<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">XMSC_DELIVERY_MODE</a>	O modo de entrega de mensagens enviadas para o destino.
<a href="#">XMSC_PRIORITY</a>	A prioridade das mensagens enviadas para o destino.
<a href="#">XMSC_RTT_MULTICAST</a>	A configuração multicast para um connection factory ou destino.
<a href="#">XMSC_TIME_TO_LIVE</a>	O tempo de vida para mensagens enviadas para o destino.
<a href="#">XMSC_WMQ_BROKER_VERSION</a>	O tipo de corretor usado pelo aplicativo para uma conexão ou para o destino.
<a href="#">XMSC_WMQ_CCSID</a>	O identificador (CCSID) do conjunto de caracteres codificados, ou página de códigos, em que as cadeias de dados de caracteres no corpo de uma mensagem estão quando o cliente XMS encaminha a mensagem para o destino.

Tabela 28. Propriedades de Destino (continuação)

Nome da propriedade	Descrição
<u>XMSC_WMQ_DUR_SUBQ</u>	<p>O nome da fila de assinantes para um assinante durável que está recebendo mensagens do destino.</p> <p><b>Nota:</b> Essa propriedade pode ser usada com a Versão 2.0 do IBM Message Service Client for .NET , mas não tem efeito para um aplicativo conectado a um gerenciador de filas do IBM WebSphere MQ 7.0 , a menos que a propriedade XMSC_WWMQ_PROVIDER_VERSION do connection factory seja configurada para um número de versão menor que 7</p>
<u>XMSC_WMQ_ENCODING</u>	<p>Como os dados numéricos no corpo de uma mensagem são representados quando o cliente XMS encaminha a mensagem para o destino.</p>
<u>XMSC_WMQ_FAIL_IF QUIESCE</u>	<p>Se as chamadas para determinados métodos falharão se o gerenciador de filas ao qual o aplicativo está conectado estiver em um estado quiesce.</p>
<u>“XMSC_WMQ_MESSAGE_BODY” na página 223</u>	<p>Essa propriedade determina se um aplicativo XMS processa o MQRFH2 de uma mensagem IBM WebSphere MQ como parte da carga útil da mensagem (ou seja, como parte do corpo da mensagem).</p>
<u>“XMSC_WMQ_MQMD_MESSAGE_CONTEXT” na página 224</u>	<p>Determina qual nível de contexto da mensagem deve ser configurado pelo aplicativo XMS . O aplicativo deve estar em execução com autoridade de contexto apropriado para esta propriedade entrar em vigor.</p>
<u>“XMSC_WMQ_MQMD_READ_ENABLED” na página 225</u>	<p>Essa propriedade determina se um aplicativo XMS pode extrair os valores de campos MQMD ou não</p>
<u>“XMSC_WMQ_MQMD_WRITE_ENABLED” na página 225</u>	<p>Esta propriedade determina se um aplicativo XMS pode ou não os valores de campos MQMD.</p>
<u>“XMSC_WMQ_READ_AHEAD_ALLOWED” na página 226</u>	<p>Essa propriedade determina se os consumidores de mensagens e os navegadores de fila têm permissão para usar leitura antecipada para obter mensagens não persistentes, não transacionais desse destino em um buffer interno antes de recebê-las.</p>
<u>“XMSC_WMQ_READ_AHEAD_CLOSE_POLICY” na página 227</u>	<p>Para mensagens que estão sendo entregues em um listener de mensagem assíncrona, essa propriedade determina o que acontece com as mensagens no buffer de leitura antecipada interno quando o consumidor de mensagens é fechado.</p>
<u>“XMSC_WMQ_RECEIVE_CCSID” na página 232</u>	<p>A propriedade de destino que configura o destino CCSID para a conversão de mensagens do gerenciador de filas. O valor é ignorado a menos que XMSC_WMQ_RECEIVE_CONVERSION seja configurado como WMQ_RECEIVE_CONVERSION_QMGR.</p>
<u>“XMSC_WMQ_RECEIVE_CONVERSION” na página 232</u>	<p>A propriedade de destino que determina se a conversão de dados será executada pelo gerenciador de filas.</p>
<u>XMSC_WMQ_TARGET_CLIENT</u>	<p>Se as mensagens enviadas para o destino contêm um cabeçalho MQRFH2.</p>

Tabela 28. Propriedades de Destino (continuação)

Nome da propriedade	Descrição
<u>XMSC_WMQ_TEMP_TOPIC_PREFIX</u>	Ao criar tópicos temporários, o XMS gera uma sequência de tópicos no formato "TEMP/TEMPTOPICPREFIX/unique_id" ou se essa propriedade contiver o valor padrão, essa sequência, "TEMP/unique_id", será gerada. Especificar um valor não vazio permite que as filas modelo específicas sejam definidas para criar as filas gerenciadas para assinantes de tópicos temporários criados sob essa conexão.
<u>XMSC_WPM_BUS_NAME</u>	Para um connection factory, o nome do barramento de integração de serviços ao qual o aplicativo se conecta ou, para um destino, o nome do barramento de integração de serviços no qual o destino existe.
<u>XMSC_WPM_TOPIC_SPACE</u>	O nome do espaço de tópico que contém o tópico.

### **Conceitos relacionados**

#### ConnectionFactories e objetos de Conexão

Um objeto ConnectionFactory fornece um modelo que um aplicativo usa para criar um objeto Connection. O aplicativo usa o objeto Connection para criar um objeto Session.

#### Conexão com um barramento de integração de serviços

Um aplicativo XMS pode se conectar a um barramento de integração de serviços WebSphere Application Server usando uma conexão TCP/IP direta ou usando HTTP sobre TCP/IP.

#### Destinos

Um aplicativo XMS usa um objeto de Destino para especificar o destino das mensagens que estão sendo enviadas e a origem de mensagens que estão sendo recebidas.

#### Curinga de destino

O XMS fornece suporte para curingas de destino, assegurando que os curingas possam ser transmitidos para o local no qual eles são necessários para correspondência. Há um esquema curinga diferente para cada tipo de servidor com o qual XMS pode trabalhar.

#### Identificadores de recursos uniformes do tópico

O URI (Identificador Uniforme de Recursos (URI) do tópico especifica o nome do tópico; ele também pode especificar uma ou mais propriedades para ele.

#### Identificadores uniformes de recursos da fila

O URI para uma fila especifica o nome da fila; ele também pode especificar uma ou mais propriedades da fila.

#### Destinos Temporários

Os aplicativos XMS podem criar e usar destinos temporários.

#### Mapeamento de Propriedades para Objetos Administrados

Para permitir que os aplicativos usem as definições de connection factory e de objeto de destino do IBM MQ JMS e do WebSphere Application Server, as propriedades recuperadas dessas definições devem ser mapeadas para as propriedades do XMS correspondentes que possam ser configuradas nas connection factories e nos destinos do XMS.

### **Tarefas relacionadas**

#### Criando Objetos Administrados

As definições de objeto ConnectionFactory e Destination que os aplicativos XMS requerem para fazer uma conexão com um servidor de sistema de mensagens devem ser criadas usando as ferramentas administrativas apropriadas.

### **Referências relacionadas**

Propriedades Necessárias para Objetos de Destino Administrados

Um aplicativo que está criando um destino deve configurar várias propriedades que o aplicativo em um objeto Destination administrado.

## Propriedades de InitialContext

Uma visão geral das propriedades do objeto InitialContext com links para informações de referência mais detalhadas.

<b>Nome da propriedade</b>	<b>Descrição</b>
<a href="#">XMSC_IC_PROVIDER_URL</a>	Usado para localizar o diretório de nomenclatura de JNDI para que o serviço de nomenclatura COS não precise estar no mesmo servidor que o serviço da web.
<a href="#">XMSC_IC_SECURITY_AUTHENTICATION</a>	Com base na Java Interface de contexto SECURITY_AUTHENTICATION Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS
<a href="#">XMSC_IC_SECURITY_CREDENTIALS</a>	Baseado na Java Interface de contexto SECURITY_CREDENTIALS Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS
<a href="#">XMSC_IC_SECURITY_PRINCIPAL</a>	Baseado na Java Interface de contexto SECURITY_PRINCIPAL Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS
<a href="#">XMSC_IC_SECURITY_PROTOCOL</a>	Com base na Java Interface de contexto SECURITY_PROTOCOL Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS
<a href="#">XMSC_IC_URL</a>	Para os contextos LDAP e FileSystem, o endereço do repositório que contém objetos administrados. Para contextos de nomenclatura COS, o endereço do serviço da web que consulta os objetos no diretório.

### Conceitos relacionados

#### [Propriedades InitialContext](#)

Os parâmetros do construtor InitialContext incluem o local do repositório de objetos administrados, fornecido como um indicador de recurso uniforme (URI). Para que um aplicativo estabeleça uma conexão com o repositório, pode ser necessário fornecer mais informações do que as informações contidas no URI.

#### [Formato de URI para contextos iniciais XMS](#)

O local do repositório de objetos administrados é fornecido como um indicador de recurso uniforme (URI). O formato do URI depende do tipo de contexto.

#### [Recuperação de Objetos Administrados](#)

XMS recupera um objeto administrado do repositório usando o endereço fornecido quando o objeto InitialContext é criado, ou nas propriedades InitialContext.

### Tarefas relacionadas

#### [Objetos InitialContext](#)

Um aplicativo deve criar um contexto inicial a ser usado para fazer uma conexão com o repositório de objetos administrados para recuperar os objetos administrados necessários.

## Propriedades de Mensagem

Uma visão geral das propriedades do objeto de Mensagem, com links para informações de referência mais detalhadas

Tabela 30. Propriedades de Mensagem

<b>Nome da propriedade</b>	<b>Descrição</b>
<u>JMS_IBM_CHARACTER_SET</u>	O identificador (CCSID) do conjunto de caracteres codificados, ou página de códigos, no qual as sequências de dados de caracteres no corpo da mensagem estão quando o cliente XMS encaminha a mensagem para seu destino desejado. No XMS , essa propriedade possui um valor numérico e é mapeado para CCSID. No entanto, esta propriedade é baseada em uma propriedade JMS portanto tem um valor de tipo string e mapas para o conjunto de caracteres Java que representa este CCSID numérico.
<u>CODIFICAÇÃO DE JMS_IBM_ENCODING</u>	Como os dados numéricos no corpo da mensagem são representados quando o cliente XMS encaminha a mensagem para seu destino desejado.
<u>JMS_IBM_EXCEPTIONMESSAGE</u>	Texto que descreve o motivo pelo qual a mensagem foi enviada para o destino de exceção. Essa propriedade é somente leitura.
<u>JMS_IBM_ExceptionProblemDestination</u>	O nome do destino em que a mensagem estava antes de a mensagem ser enviada para o destino de exceção.
<u>JMS_IBM_EXCEPTIONREASON</u>	Um código de razão que indica a razão pela qual a mensagem foi enviada para o destino de exceção.
<u>JMS_IBM_EXCEPTIONTIMESTAMP</u>	O horário em que a mensagem foi enviada para o destino de exceção.
<u>JMS_IBM_FEEDBACK</u>	Um código que indica a natureza de uma mensagem de relatório.
<u>FORMATO JMS_IBM_FORMAT</u>	A natureza dos dados do aplicativo na mensagem
<u>JMS_IBM_LAST_MSG_IN_GROUP</u>	Indicar se a mensagem é a última mensagem em um grupo de mensagens.
<u>JMS_IBM_MSGTYPE</u>	O tipo de mensagem.
<u>JMS_IBM_PUTAPPLTYPE</u>	O tipo de aplicativo que enviou a mensagem.
<u>JMS_IBM_PUTDATE</u>	A data em que a mensagem foi enviada.
<u>JMS_IBM_PUTTIME</u>	O horário em que a mensagem foi enviada.
<u>JMS_IBM_REPORT_COA</u>	Solicitar mensagens de relatório 'confirmar na chegada', especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.
<u>JMS_IBM_REPORT_COD</u>	Solicitar mensagens de relatório 'confirmar na entrega', especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.
<u>JMS_IBM_REPORT_DISCARD_MSG</u>	Solicitar que a mensagem seja descartada se não puder ser entregue a seu destino desejado.
<u>JMS_IBM_REPORT_EXCEPTION</u>	Solicitar mensagens de relatório de exceção, especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.

Tabela 30. Propriedades de Mensagem (continuação)

Nome da propriedade	Descrição
<u>JMS_IBM_REPORT_EXPIRATION</u>	Solicitar mensagens de relatório de expiração, especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.
<u>JMS_IBM_REPORT_NAN</u>	Solicitar mensagens de relatório de notificação de ação negativa.
<u>JMS_IBM_REPORT_PAN</u>	Solicitar mensagens de relatório de notificação de ação positiva.
<u>JMS_IBM_Report_Pass_Correl_ID</u>	Solicitar que o identificador de correlação de qualquer mensagem de relatório ou de resposta seja o mesmo que o identificador de correlação da mensagem original.
<u>JMS_IBM_REPORT_PASS_MSG_ID</u>	Solicitar que o identificador de mensagem de qualquer mensagem de relatório ou resposta seja o mesmo que o identificador de mensagem da mensagem original.
<u>JMS_IBM_RETAIN</u>	Configurar essa propriedade indica ao gerenciador de filas para tratar uma mensagem como Publicação retida.
<u>JMS_IBM_SYSTEM_MESSAGEID</u>	Um identificador que identifica a mensagem com exclusividade dentro do barramento de integração de serviços. Essa propriedade é somente leitura.
<u>JMSX_APPID</u>	O nome do aplicativo que enviou a mensagem.
<u>JMSX_DELIVERY_COUNT</u>	O número de tentativas de entregar a mensagem.
<u>JMSX_GROUPID</u>	O identificador do grupo de mensagens ao qual a mensagem pertence.
<u>JMSX_GROUPSEQ</u>	O número de sequência da mensagem dentro de um grupo de mensagens.
<u>JMSX_USERID</u>	O identificador de usuários associado ao aplicativo que enviou a mensagem.

### Propriedades JMS\_IBM\_MQMD\*

IBM Message Service Client for .NET permite que aplicativos clientes leiam / gravem campos MQMD usando APIs. Ele também permite o acesso aos dados da mensagens do MQ Por padrão, o acesso ao MQMD é desativado e deve ser ativado explicitamente pelo aplicativo usando as propriedades de Destino XMSC\_WMQ\_MQMD\_WRITE\_ENABLED e XMSC\_WMQ\_MQMD\_READ\_ENABLED Essas duas propriedades são independentes entre si.

Todos os campos MQMD, exceto StrucId e Version são expostos como propriedades adicionais do objeto de Mensagem e são prefixados JMS\_IBM\_MQMD.

As propriedades JMS\_IBM\_MQMD\* têm precedência mais alta sobre outras propriedades como JMS\_IBM\* descritas na tabela anterior.

### Enviando mensagens

Todos os campos MQMD, exceto StrucId e Version, são representados. Essas propriedades referem-se apenas aos campos MQMD; quando uma propriedade ocorre tanto no MQMD quanto no cabeçalho MQRFH2, a versão no MQRFH2 não é configurada nem extraída. Qualquer uma dessas

propriedades pode ser configurada, exceto JMS\_IBM\_MQMD\_BackoutCount. Qualquer valor configurado para JMS\_IBM\_MQMD\_BackoutCount é ignorado.

Se uma propriedade tiver um comprimento máximo e você fornecer um valor que é muito longo, o valor será truncado.

Para determinadas propriedades, deve-se também configurar a propriedade XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT no objeto de Destino O aplicativo deve estar em execução com autoridade de contexto apropriado para esta propriedade entrar em vigor. Se não configurar XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT para um valor apropriado, o valor da propriedade será ignorado. Se você configurar XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT para um valor apropriado, mas não tiver autoridade de contexto suficiente para o gerenciador de filas, uma exceção será emitida.. Propriedades que requerem valores específicos de XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT são as seguintes.

As propriedades a seguir requerem que XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT seja configurado como XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT ou XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT:

- JMS\_IBM\_MQMD\_UserIdentifier
- JMS\_IBM\_MQMD\_AccountingToken
- JMS\_IBM\_MQMD\_ApplIdentityData

As seguintes propriedades requerem que XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT seja configurado como 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

## Como receber mensagens

Todas essas propriedades estarão disponíveis em uma mensagem recebida se a propriedade XMSC\_WMQ\_MQMD\_READ\_ENABLED estiver configurada como true, independentemente das propriedades reais que o aplicativo de produção configurou como true. Um aplicativo não pode modificar as propriedades de uma mensagem recebida a menos que todas as propriedades sejam limpas primeiro, de acordo com a especificação JMS. A mensagem recebida pode ser transmitida sem modificar as propriedades.

**Nota:** Se seu aplicativo receber uma mensagem de um destino com a propriedade XMSC\_WMQ\_MQMD\_READ\_ENABLED configurada como true e encaminhá-la para um destino com XMSC\_WMQ\_MQMD\_WRITE\_ENABLED configurado como true, isso resultará em todos os valores do campo MQMD da mensagem recebida sendo copiados na mensagem encaminhada. Tabela de propriedades

<i>Tabela 31. Propriedades do objeto de Mensagem que representa os campos MQMD</i>		
<b>Propriedade</b>	<b>Descrição</b>	<b>Tipo</b>
JMS_IBM_MQMD_Report	Opções para as mensagens de relatório	System.Int32
JMS_IBM_MQMD_MsgType	Tipo de Mensagem	System.Int32
JMS_IBM_MQMD_Expiry	Tempo de vida da mensagem	System.Int32
JMS_IBM_MQMD_Feedback	Feedback ou código de razão	System.Int32
JMS_IBM_MQMD_Encoding	Codificação numérica de dados da mensagem	System.Int32

<i>Tabela 31. Propriedades do objeto de Mensagem que representa os campos MQMD (continuação)</i>		
<b>Propriedade</b>	<b>Descrição</b>	<b>Tipo</b>
JMS_IBM_MQMD_CodedCharSetId	Identificador do conjunto de caracteres de dados da mensagem	System.Int32
JMS_IBM_MQMD_Format	Nome do formato dos dados da mensagem	System.String
JMS_IBM_MQMD_PRIORITY <b>Nota:</b> Se você designar um valor para JMS_IBM_MQMD_PRIORITY que não esteja no intervalo de 0 a 9, esse valor violará a especificação JMS.	Prioridade da mensagem	System.Int32
JMS_IBM_MQMD_Persistence	Persistência de mensagem	System.Int32
JMS_IBM_MQMD_MSGID <b>Nota:</b> A especificação JMS indica que o ID de mensagem deve ser configurado pelo provedor JMS e que deve ser exclusivo ou nulo. Se você designar um valor para JMS_IBM_MQMD_MsgId, esse valor será copiado para o JMSMessageID. Portanto, ele não é configurado pelo provedor JMS e pode não ser exclusivo: este valor viola a especificação JMS..	ID da Mensagem	matriz de byte <b>Nota:</b> O uso de propriedades de matriz de bytes em uma mensagem viola a especificação JMS
JMS_IBM_MQMD_CORRELID <b>Nota:</b> Se você designar um valor para JMS_IBM_MQMD_CORRELID que inicia com a sequência 'ID:', esse valor viola a especificação JMS.	Identificador de correlação	matriz de byte <b>Nota:</b> O uso de propriedades de matriz de bytes em uma mensagem viola a especificação JMS
JMS_IBM_MQMD_BackoutCount	contador de backout	System.Int32
JMS_IBM_MQMD_ReplyToQ	Nome da fila de resposta	System.String
JMS_IBM_MQMD_ReplyToQMgr	Nome do gerenciador de filas de resposta	System.String
JMS_IBM_MQMD_UserIdentifier	Identificador de usuário	System.String
JMS_IBM_MQMD_AccountingToken	Símbolo de contabilidade	matriz de byte <b>Nota:</b> O uso de propriedades de matriz de bytes em uma mensagem viola a especificação JMS
JMS_IBM_MQMD_ApplIdentityData	dados do aplicativo relacionados à identidade	System.String
JMS_IBM_MQMD_PutApplType	Tipo de aplicativo que coloca a mensagem	System.Int32
JMS_IBM_MQMD_PutApplName	Nome do aplicativo que colocou a mensagem	System.String
JMS_IBM_MQMD_PutDate	Data quando a mensagem foi colocada	System.String

*Tabela 31. Propriedades do objeto de Mensagem que representa os campos MQMD (continuação)*

Propriedade	Descrição	Tipo
JMS_IBM_MQMD_PutTime	Hora quando a mensagem foi colocada	System.String
JMS_IBM_MQMD_ApplOriginData	Os dados do aplicativo relacionados à origem	System.String
JMS_IBM_MQMD_GroupId	Identificador de grupo	matriz de byte <b>Nota:</b> O uso de propriedades de matriz de bytes em uma mensagem viola a especificação JMS
JMS_IBM_MQMD_MsgSeqNumber	Número de sequência da mensagem local no grupo	System.Int32
JMS_IBM_MQMD_Offset	Deslocamento dos dados na mensagem física a partir do início da mensagem lógica	System.Int32
JMS_IBM_MQMD_MsgFlags	Sinalizadores de mensagem	System.Int32
JMS_IBM_MQMD_OriginalLength	Comprimento da mensagem original	System.Int32

Consulte [MQMD](#) para obter detalhes adicionais..

## Examples

Este exemplo resulta em uma mensagem sendo colocada em uma fila ou em um tópico com MQMD.UserIdentifier configurado como "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
// ...
```

É necessário configurar XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT antes de configurar JMS\_IBM\_MQMD\_USERIDENTIFIER. Para obter mais informações sobre o uso de XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT, consulte Propriedades do objeto Message.

Da mesma forma, é possível extrair o conteúdo dos campos do MQMD configurando XMSC\_WMQ\_MQMD\_READ\_ENABLED como true antes de receber uma mensagem e, em seguida, usando os métodos get da mensagem, como a propriedade getString. As propriedades recebidas são somente leitura.

Este exemplo resulta no campo de valor contendo o valor do MQMD MQMD.ApplIdentityData campo de uma mensagem obtido de uma fila ou de um tópico.

```
// 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 required MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

## Propriedades de MessageConsumer

Uma visão geral das propriedades do objeto MessageConsumer com links para informações de referência mais detalhadas.

Nome da propriedade	Descrição
<a href="#">XMSC_IS_SUBSCRIPTION_MULTICAST</a>	Indica se as mensagens estão sendo entregues para o consumidor de mensagens usando WebSphere MQ Multicast Transport Essa propriedade é somente leitura.
<a href="#">XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST</a>	Indica se as mensagens estão sendo entregues ao consumidor de mensagem usando WebSphere MQ Multicast Transport com uma qualidade de serviço confiável. Essa propriedade é somente leitura.

Consulte [.As propriedades NET de IMessageConsumer](#) para obter mais detalhes

## Propriedades do MessageProducer

Uma visão geral das propriedades do objeto MessageProducer , com links para informações de referência mais detalhadas.

Consulte [.Propriedades NET de IMessageProducer](#) para obter mais detalhes.

## Propriedades da Sessão.

Uma visão geral das propriedades do objeto de Sessão, com links para informações de referência mais detalhadas

Consulte [.Propriedades NET de ISession](#) para obter mais detalhes

## Definições de propriedades

Esta seção fornece uma definição de cada propriedade de objeto

Cada definição de propriedade inclui as seguintes informações:

- O tipo de dados da propriedade
- Os tipos de objeto que possuem a propriedade
- Para uma propriedade de Destino, o nome que pode ser usado em um URI (Identificador Uniforme de Recursos)
- Uma descrição mais detalhada da propriedade
- Os valores válidos da propriedade

- O valor padrão da propriedade

As propriedades cujos nomes começam com um dos seguintes prefixos são relevantes apenas para o tipo de conexão especificado:

#### **XMSC\_RTT**

As propriedades são relevantes apenas para uma conexão em tempo real com um broker Os nomes das propriedades são definidos como constantes nomeadas no arquivo de cabeçalho `xmsc_rtt.h`

#### **XMSC\_WMQ**

As propriedades são relevantes somente quando um aplicativo se conecta a um gerenciador de filas do WebSphere MQ Os nomes das propriedades são definidos como constantes nomeadas no arquivo de cabeçalho `xmsc_wmq.h`

#### **XMSC\_WPM**

As propriedades são relevantes apenas quando um aplicativo se conecta a um barramento de integração de serviços do WebSphere Os nomes das propriedades são definidos como constantes nomeadas no arquivo de cabeçalho `xmsc_wpm.h`

Salvo indicação em contrário em suas definições, as propriedades restantes são relevantes para todos os tipos de conexão. Os nomes das propriedades são definidos como constantes nomeadas no arquivo de cabeçalho `xmsc.h` Propriedades cujos nomes começam com o prefixo JMSX são JMS propriedades definidas de uma mensagem e propriedades cujos nomes começam com o prefixo JMS\_IBM são IBM propriedades definidas de uma mensagem. Para obter mais informações sobre as propriedades das mensagens, consulte [“Propriedades de uma mensagem XMS” na página 72](#)

A menos que indicado de outra forma em sua definição, cada propriedade é relevante nos domínios ponto-a-ponto e Publicação/Assinatura .

Um aplicativo pode obter e configurar o valor de qualquer propriedade, a não ser que a propriedade seja designada como somente leitura.

### **JMS\_IBM\_CHARACTER\_SET**

#### **Tipo de dado:**

System.Int32

#### **Propriedade de:**

Mensagem

O identificador (CCSID) do conjunto de caracteres codificados, ou página de códigos, no qual as sequências de dados de caracteres no corpo da mensagem estão quando o cliente XMS encaminha a mensagem para seu destino desejado. No XMS , essa propriedade possui um valor numérico e é mapeado para CCSID. No entanto, esta propriedade é baseada em uma propriedade JMS portanto tem um valor de tipo string e mapas para o conjunto de caracteres Java que representa este CCSID numérico. Essa propriedade substitui qualquer CCSID especificado para o destino pela propriedade XMSC\_WMQ\_CC SID .

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

### **CODIFICAÇÃO DE JMS\_IBM\_ENCODING**

#### **Tipo de dado:**

System.Int32

#### **Propriedade de:**

Mensagem

Como os dados numéricos no corpo da mensagem são representados quando o cliente XMS encaminha a mensagem para seu destino desejado. Essa propriedade substitui qualquer codificação especificada para o destino pela propriedade XMSC\_WMQ\_ENCODING A propriedade especifica a representação de números inteiros binários, números inteiros decimais empacotados e números de vírgula flutuante..

Os valores válidos da propriedade são os mesmos que os valores que podem ser especificados no campo **Encoding** de um descritor de mensagem

Um aplicativo pode usar as seguintes constantes nomeadas para configurar a propriedade:

<b>Constante nomeada</b>	<b>Significado</b>
MQENC_INTEGER_NORMAL	Codificação de número inteiro normal
MQENC_INTEGER_REVERSED	Codificação de número inteiro revertida
MQENC_DECIMAL_NORMAL	Codificação decimal compactada normal
MQENC_DECIMAL_REVERSED	Codificação decimal compactada revertida
MQENC_FLOAT_IEEE_NORMAL	Codificação de ponto flutuante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codificação de ponto flutuante IEEE revertida
MQENC_FLOAT_S390	Codificação de ponto flutuante da arquitetura z/OS
MQENC_NATIVE	Codificação de máquina nativa

Para formar um valor para a propriedade, o aplicativo pode incluir três dessas constantes da seguinte forma:

- Uma constante cujo nome começa com MQENC\_INTEGER, para especificar a representação de inteiros binários
- Uma constante cujo nome começa com MQENC\_DECIMAL, para especificar a representação de números inteiros decimais compactados
- Uma constante cujo nome começa com MQENC\_FLOAT, para especificar a representação de números de ponto flutuante

Como alternativa, o aplicativo pode configurar a propriedade para MQENC\_NATIVE, cujo valor é dependente de ambiente.

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

## ***JMS\_IBM\_EXCEPTIONMESSAGE***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

Mensagem

Texto que descreve o motivo pelo qual a mensagem foi enviada para o destino de exceção. Essa propriedade é somente leitura.

Essa propriedade é relevante apenas quando um aplicativo se conecta a um barramento de integração de serviços e recebe uma mensagem de um destino de exceções

## ***JMS\_IBM\_ExceptionProblemDestination***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

Mensagem

O nome do destino em que a mensagem estava antes de a mensagem ser enviada para o destino de exceção.

Essa propriedade é relevante apenas quando um aplicativo se conecta a um barramento de integração de serviços e recebe uma mensagem de um destino de exceções

## ***JMS\_IBM\_EXCEPTIONREASON***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Um código de razão que indica a razão pela qual a mensagem foi enviada para o destino de exceção.

Essa propriedade é relevante apenas quando um aplicativo se conecta a um barramento de integração de serviços e recebe uma mensagem de um destino de exceções

## ***JMS\_IBM\_EXCEPTIONTIMESTAMP***

**Tipo de dado:**

System.Int64

**Propriedade de:**

Mensagem

O horário em que a mensagem foi enviada para o destino de exceção.

O tempo é expresso em milissegundos desde 00:00:00 GMT de 1 de janeiro de 1970.

Essa propriedade é relevante apenas quando um aplicativo se conecta a um barramento de integração de serviços e recebe uma mensagem de um destino de exceções

## ***JMS\_IBM\_FEEDBACK***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Um código que indica a natureza de uma mensagem de relatório.

Os valores válidos da propriedade são os códigos de feedback e de razão que podem ser especificados no campo **Feedback** de um descritor de mensagens.

Por padrão, a propriedade não é configurada

## ***FORMATO JMS\_IBM\_FORMAT***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

A natureza dos dados do aplicativo na mensagem

Os valores válidos da propriedade são os mesmos que os valores que podem ser especificados no campo **Format** de um descritor de mensagem

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

## ***JMS\_IBM\_LAST\_MSG\_IN\_GROUP***

**Tipo de dado:**

System.Boolean

**Propriedade de:**

Mensagem

Indicar se a mensagem é a última mensagem em um grupo de mensagens.

Configure a propriedade como true se a mensagem for a última mensagem em um grupo de mensagens.. Caso contrário, configure a propriedade como false, ou não configure a propriedades. Por padrão, a propriedade não é configurada

O valor true corresponde à sinalização de status MQMF\_LAST\_MSG\_IN\_GROUP, que pode ser especificada no campo **MsgFlags** de um descritor de mensagens.

Essa propriedade é ignorada no domínio Publicação/Assinatura e não é relevante quando um aplicativo se conecta a um barramento de integração de serviços..

### ***JMS\_IBM\_MSGTYPE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

O tipo de mensagem.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
MQMT_DATAGRAM	A mensagem é uma que não requer resposta.
MQMT_REQUEST	A mensagem é aquela que requer uma resposta.
MQMT_REPLY	A mensagem é uma mensagem de resposta
MQMT_REPORT	A mensagem é uma mensagem de relatório

Esses valores correspondem aos tipos de mensagens que podem ser especificados no campo **MsgType** de um descritor de mensagens

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

### ***JMS\_IBM\_PUTAPPLTYPE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

O tipo de aplicativo que enviou a mensagem.

Os valores válidos da propriedade são os tipos de aplicativos que podem ser especificados no campo **PutApp1Type** de um descritor de mensagens

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

### ***JMS\_IBM\_PUTDATE***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

A data em que a mensagem foi enviada.

Os valores válidos da propriedade são os mesmos que os valores que podem ser especificados no campo **PutDate** de um descritor de mensagem

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

### ***JMS\_IBM\_PUTTIME***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

O horário em que a mensagem foi enviada.

Os valores válidos da propriedade são os mesmos que os valores que podem ser especificados no campo **PutTime** de um descritor de mensagem

Por padrão, a propriedade não é configurada

Essa propriedade não é relevante quando um aplicativo se conecta a um barramento de integração de serviços

### ***JMS\_IBM\_REPORT\_COA***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório 'confirmar na chegada', especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
MQRO_COA	Solicite mensagens de relatório 'confirmar na chegada', sem dados do aplicativo da mensagem original incluídos em uma mensagem de relatório
MQRO_COA_WITH_DATA	Solicite 'confirmar na chegada' mensagens de relatório, com os primeiros 100 bytes de dados do aplicativo da mensagem original incluídos em uma mensagem de relatório
MQRO_COA_WITH_FULL_DATA	Solicite mensagens de relatório 'confirmar na chegada', com todos os dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Esses valores correspondem às opções do relatório que podem ser especificadas no campo **Report** de um descritor de mensagens Para obter mais informações sobre essas opções, consulte [Report \(MQLONG\)](#).

Por padrão, a propriedade não é configurada

### ***JMS\_IBM\_REPORT\_COD***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório 'confirmar na entrega', especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
MQRO_COD	Solicite 'confirmar na entrega' mensagens de relatório, sem dados do aplicativo da mensagem original incluída em uma mensagem de relatório
MQRO_COD_WITH_DATA	Solicite 'confirmar na entrega' mensagens de relatório, com os primeiros 100 bytes de dados do aplicativo da mensagem original incluídos em uma mensagem de relatório
MQRO_COD_WITH_FULL_DATA	Solicite 'confirmar na entrega' mensagens de relatório, com todos os dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Esses valores correspondem às opções de relatório que podem ser especificadas no campo **Report** de um descritor de mensagens

Por padrão, a propriedade não é configurada

### ***JMS\_IBM\_REPORT\_DISCARD\_MSG***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar que a mensagem seja descartada se não puder ser entregue a seu destino desejado.

Configure a propriedade para MQRO\_DISCARD\_MSG para solicitar que a mensagem seja descartada se não puder ser entregue para seu destino desejado. Se você requerer que a mensagem seja colocada em uma fila de devoluções ou enviada para um destino de exceções, não configure a propriedade. Por padrão, a propriedade não é configurada

O valor MQRO\_DISCARD\_MSG corresponde a uma opção de relatório que pode ser especificada no campo **Report** de um descritor de mensagens

### ***JMS\_IBM\_REPORT\_EXCEPTION***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório de exceção, especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
MQRO_EXCEPTION	Mensagens de relatório de exceção de solicitação, sem dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

**Valor válido**

MORO\_EXCEPTION\_WITH\_DATA

MORO\_EXCEPTION\_WITH\_FULL\_DATA

**Significado**

Mensagens de relatório de exceção de solicitação, com os primeiros 100 bytes de dados do aplicativo da mensagem original incluídos em uma mensagem de relatório.

Mensagens de relatório de exceção de solicitação, com todos os dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Esses valores correspondem às opções de relatório que podem ser especificadas no campo **Report** de um descritor de mensagens

Por padrão, a propriedade não é configurada

**JMS\_IBM\_REPORT\_EXPIRATION****Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório de expiração, especificando quantos dados do aplicativo da mensagem original devem ser incluídos em uma mensagem de relatório.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

MORO\_EXPIRATION

MORO\_EXPIRATION\_WITH\_DATA

MORO\_EXPIRATION\_WITH\_FULL\_DATA

**Significado**

Mensagens de relatório de expiração de solicitação, sem dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Mensagens de relatório de expiração de solicitação, com os primeiros 100 bytes de dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Mensagens de relatório de expiração de solicitação, com todos os dados do aplicativo da mensagem original incluídos em uma mensagem de relatório

Esses valores correspondem às opções de relatório que podem ser especificadas no campo **Report** de um descritor de mensagens

Por padrão, a propriedade não é configurada

**JMS\_IBM\_REPORT\_NAN****Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório de notificação de ação negativa.

Configure a propriedade para MORO\_NAN para solicitar mensagens de notificação de ação negativa. Se você não precisar de mensagens de notificação de ação negativa, não configure a propriedade. Por padrão, a propriedade não é configurada

O valor MQRO\_NAN corresponde a uma opção de relatório que pode ser especificada no campo **Report** de um descritor de mensagens.

### ***JMS\_IBM\_REPORT\_PAN***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar mensagens de relatório de notificação de ação positiva.

Configure a propriedade como MQRO\_PAN para solicitar mensagens de relatório de notificação de ação positiva. Se você não precisar de mensagens de notificação de ação positiva, não configure a propriedade. Por padrão, a propriedade não é configurada.

O valor MQRO\_PAN corresponde a uma opção de relatório que pode ser especificada no campo **Report** de um descritor de mensagens.

### ***JMS\_IBM\_Report\_Pass\_Correl\_ID***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar que o identificador de correlação de qualquer mensagem de relatório ou de resposta seja o mesmo que o identificador de correlação da mensagem original.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

MQRO\_PASS\_CORREL\_ID

**Significado**

Solicitar que o identificador de correlação de qualquer mensagem de relatório ou de resposta seja o mesmo que o identificador de correlação da mensagem original.

MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID

Solicite que o identificador de correlação de qualquer mensagem de relatório ou de resposta seja igual ao identificador de mensagem da mensagem original.

Esses valores correspondem às opções do relatório, que podem ser especificadas no campo **Report** de um descritor de mensagem.

O valor padrão da propriedade é MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID.

### ***JMS\_IBM\_REPORT\_PASS\_MSG\_ID***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Solicitar que o identificador de mensagem de qualquer mensagem de relatório ou resposta seja o mesmo que o identificador de mensagem da mensagem original.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
MQRO_PASS_MSG_ID	Solicitar que o identificador de mensagem de qualquer mensagem de relatório ou resposta seja o mesmo que o identificador de mensagem da mensagem original.
MQRO_NEW_MSG_ID	Solicite que um novo identificador de mensagem seja gerado para cada mensagem de relatório ou resposta.

Esses valores correspondem às opções do relatório que podem ser especificadas no campo [Relatório](#) de um descritor de mensagens

O valor padrão da propriedade é MQRO\_NEW\_MSG\_ID.

### ***JMS\_IBM\_RETAIN***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

Configurar essa propriedade indica ao gerenciador de filas para tratar uma mensagem como Publicação retida. Quando um assinante recebe mensagens de tópicos, ele pode receber mensagens adicionais imediatamente após a assinatura, além das mensagens recebidas em liberações anteriores. Essas mensagens são as publicações opcionais retidas para os tópicos inscritos. Para cada tópico correspondente à assinatura, se houver uma publicação retida, a publicação será disponibilizada para entrega ao consumidor de mensagens de assinatura.

RETAIN\_PUBLICATION é o único valor válido para esta propriedade. Por padrão, esta propriedade não é definida.

**Nota:** Esta propriedade é relevante apenas no domínio de publicação / assinatura

### ***JMS\_IBM\_SYSTEM\_MESSAGEID***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

Um identificador que identifica a mensagem com exclusividade dentro do barramento de integração de serviços. Essa propriedade é somente leitura.

Esta propriedade é relevante apenas quando um aplicativo se conecta a um barramento de integração de serviços

### ***JMSX\_APPID***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

O nome do aplicativo que enviou a mensagem.

Essa propriedade é a propriedade definida JMS com o JMS name JMSXAppID. Para obter mais informações sobre a propriedade, consulte a *Java Message Service Specification, Versão 1.1*.

Por padrão, a propriedade não é configurada

Essa propriedade não é válida para uma conexão em tempo real com um broker

## ***JMSX\_DELIVERY\_COUNT***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

O número de tentativas de entregar a mensagem.

Essa propriedade é a propriedade definida pelo JMS com o JMS name JMSXDeliveryCount Para obter mais informações sobre a propriedade, consulte a *Java Message Service Specification, Versão 1.1*.

Por padrão, a propriedade não é configurada

Essa propriedade não é válida para uma conexão em tempo real com um broker

## ***JMSX\_GROUPID***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

O identificador do grupo de mensagens ao qual a mensagem pertence.

Essa propriedade é a propriedade definida JMS com o JMS nome JMSXGroupID. Para obter mais informações sobre a propriedade, consulte a *Java Message Service Specification, Versão 1.1*.

Por padrão, a propriedade não é configurada

Essa propriedade não é válida para uma conexão em tempo real com um broker

## ***JMSX\_GROUPSEQ***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Mensagem

O número de sequência da mensagem dentro de um grupo de mensagens.

Essa propriedade é a propriedade definida JMS com o JMS nome JMSXGroupSeq. Para obter mais informações sobre a propriedade, consulte a *Java Message Service Specification, Versão 1.1*.

Por padrão, a propriedade não é configurada

Essa propriedade não é válida para uma conexão em tempo real com um broker

## ***JMSX\_USERID***

**Tipo de dado:**

Sequência

**Propriedade de:**

Mensagem

O identificador de usuários associado ao aplicativo que enviou a mensagem.

Essa propriedade é a propriedade definida JMS com o JMS name JMSXUserID. Para obter mais informações sobre a propriedade, consulte a *Java Message Service Specification, Versão 1.1*.

Por padrão, a propriedade não é configurada

Essa propriedade não é válida para uma conexão em tempo real com um broker

## ***XMSC\_ASYNC\_EXCEPÇÕES***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: ASYNCEXCEPTION

Nome abreviado da ferramenta de administração JMS: AEX

Essa propriedade determina se o XMS informa um ExceptionListener apenas quando uma conexão é quebrada ou quando qualquer exceção ocorre de forma assíncrona para uma chamada da API do XMS. Essa propriedade se aplica a todas as Conexões criadas por meio dessa ConnectionFactory que possui um ExceptionListener registrado.

Os valores válidos para essa propriedade são:

### ***XMSC\_ASYNC\_EXCEPTIONS\_ALL***

Qualquer exceção detectada de forma assíncrona, fora do escopo de uma chamada de API síncrona, e todas as exceções de conexão interrompida são enviadas para o ExceptionListener

### ***XMSC\_ASYNC\_EXCEPTIONS\_CONNECTIONBROKEN***

Apenas exceções indicando uma conexão interrompida são enviadas para o ExceptionListener.

Quaisquer outras exceções que ocorram durante o processamento assíncrono não são relatadas para o ExceptionListenere, portanto, o aplicativo não é informado sobre essas exceções

Por padrão, essa propriedade é configurada para XMSC\_ASYNC\_EXCEPTIONS\_ALL

## ***XMSC\_CLIENT\_ID***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: CLIENTID

Nome abreviado da ferramenta de administração JMS: CID

O identificador do cliente para uma conexão.

Um identificador de cliente é usado apenas para suportar assinaturas duráveis no domínio Publicação/ Assinatura e é ignorado no domínio ponto-a-ponto . Para obter mais informações sobre a configuração de identificadores de cliente, consulte [“ConnectionFactoryes e objetos de Conexão” na página 21](#)

Essa propriedade não é relevante para uma conexão em tempo real com um broker

## ***XMSC\_CONNECTION\_TYPE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O tipo de servidor de mensagens para o qual um aplicativo se conecta.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
XMSC_CT_RTT	Uma conexão em tempo real com um corretor.
XMSC_CT_WMQ	Uma conexão com um gerenciador de filas do WebSphere MQ

**Valor válido**

XMSC\_CT\_WPM

**Significado**

Uma conexão com um barramento de integração de serviços do WebSphere

Por padrão, a propriedade não é configurada

***XMSC\_DELIVERY\_MODE*****Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

**Nome usado em um URI:**

persistência (para um destino WebSphere MQ)

deliveryMode (para um WebSphere destino do provedor de sistemas de mensagens padrão)

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: PERSISTENCE

Nome abreviado da ferramenta de administração JMS: PER

O modo de entrega de mensagens enviadas para o destino.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

XMSC\_DELIVERY\_NOT\_PERSISTENT

**Significado**

Uma mensagem enviada para o destino é não persistente O modo de entrega padrão do produtor de mensagem, ou qualquer modo de entrega especificado na chamada Enviar, é ignorado. Se o destino for uma fila WebSphere MQ, o valor do atributo da fila *DefPersistence* também será ignorado.

XMSC\_DELIVERY\_PERSISTENT

Uma mensagem enviada para o destino é persistente O modo de entrega padrão do produtor de mensagem, ou qualquer modo de entrega especificado na chamada Enviar, é ignorado. Se o destino for uma fila WebSphere MQ, o valor do atributo da fila *DefPersistence* também será ignorado.

XMSC\_DELIVERY\_AS\_APP

Uma mensagem enviada para o destino tem o modo de entrega especificado na chamada Enviar. Se a chamada Enviar especificar nenhum modo de entrega, o modo de entrega padrão do produtor de mensagem será usado no lugar. Se o destino for uma fila WebSphere MQ, o valor do atributo da fila *DefPersistence* será ignorado.

**Valor válido**

XMSC\_DELIVERY\_AS\_DEST

**Significado**

Se o destino for uma fila WebSphere MQ , uma mensagem colocada na fila terá o modo de entrega especificado pelo valor do atributo da fila *DefPersistence*. O modo de entrega padrão do produtor de mensagem, ou qualquer modo de entrega especificado na chamada Enviar, é ignorado.

Se o destino não for uma fila WebSphere MQ , o significado será o mesmo de XMSC\_DELIVERY\_AS\_APP.

O valor-padrão é XMSC\_DELIVERY\_AS\_APP.

***XMSC\_IC\_PROVIDER\_URL*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Usado para localizar o diretório de nomenclatura de JNDI para que o serviço de nomenclatura COS não precise estar no mesmo servidor que o serviço da web.

***XMSC\_IC\_SECURITY\_AUTHENTICATION*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Com base na Java Interface de contexto SECURITY\_AUTHENTICATION Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS

***XMSC\_IC\_SECURITY\_CREDENTIALS*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Baseado na Java Interface de contexto SECURITY\_CREDENTIALS Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS

***XMSC\_IC\_SECURITY\_PRINCIPAL*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Baseado na Java Interface de contexto SECURITY\_PRINCIPAL Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS

***XMSC\_IC\_SECURITY\_PROTOCOL*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Com base na Java Interface de contexto SECURITY\_PROTOCOL . Essa propriedade é aplicável apenas ao contexto de nomenclatura do COS

***XMSC\_IC\_URL*****Tipo de dado:**

Sequência

**Propriedade de:**

InitialContext

Para os contextos LDAP e FileSystem, o endereço do repositório que contém objetos administrados.

Para contextos de nomenclatura COS, o endereço do serviço da web que consulta os objetos no diretório.

***XMSC\_IS\_SUBSCRIPTION\_MULTICAST*****Tipo de dado:**

System.Boolean

**Propriedade de:**

MessageConsumer

Indica se as mensagens estão sendo entregues para o consumidor de mensagens usando WebSphere MQ Multicast Transport Essa propriedade é somente leitura.

O valor da propriedade será true se as mensagens estiverem sendo entregues ao consumidor de mensagens usando WebSphere MQ Multicast Transport. Caso contrário, o valor será false.

Essa propriedade é relevante apenas para uma conexão em tempo real com um broker

***XMSC\_IS\_SUBSCRIPTION\_RELIABLE\_MULTICAST*****Tipo de dado:**

System.Boolean

**Propriedade de:**

MessageConsumer

Indica se as mensagens estão sendo entregues ao consumidor de mensagem usando WebSphere MQ Multicast Transport com uma qualidade de serviço confiável. Essa propriedade é somente leitura.

O valor da propriedade será true se as mensagens estiverem sendo entregues para o consumidor de mensagens usando WebSphere MQ Multicast Transport com uma qualidade de serviço confiável Caso contrário, o valor será false.

Essa propriedade é relevante apenas para uma conexão em tempo real com um broker

***XMSC\_JMS\_MAJOR\_VERSION*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionMetaData

O número da versão principal da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.

***XMSC\_JMS\_MINOR\_VERSION*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionMetaData

O número da versão secundária da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.

***XMSC\_JMS\_VERSION*****Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionMetaData

O identificador de versão da especificação JMS na qual XMS é baseado. Essa propriedade é somente leitura.

***XMSC\_MAJOR\_VERSION*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionMetaData

O número da versão do cliente do XMS Essa propriedade é somente leitura.

***XMSC\_MINOR\_VERSION*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionMetaData

O número da liberação do cliente do XMS Essa propriedade é somente leitura.

***XMSC\_PASSWORD*****Tipo de dado:**

matriz de byte

**Propriedade de:**

ConnectionFactory

Uma senha que pode ser usada para autenticar o aplicativo quando ele tenta se conectar a um servidor de mensagens. A senha é usada com a propriedade XMSC\_USERID

Por padrão, a propriedade não é configurada

 Se você estiver se conectando ao WebSphere MQ on Multiplataformase configurar a propriedade XMSC\_USERID do connection factory, ele deverá corresponder ao **userid** do usuário conectado. Se você não configurar essas propriedades, o gerenciador de filas usará o **userid** do usuário conectado por padrão. Se você precisar de autenticação no nível de conexão adicional de usuários individuais, poderá gravar uma saída de autenticação de cliente que está configurada em WebSphere MQ. É possível aprender mais sobre a criação de uma saída de autenticação de cliente no tópico Autenticação no manual de Clientes do WebSphere MQ

Para autenticar o usuário ao conectar ao WebSphere MQ on z/OS , é necessário usar uma saída de segurança.

***XMSC\_PRIORITY*****Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

**Nome usado em um URI:**

priority

A prioridade das mensagens enviadas para o destino.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
Um número inteiro no intervalo 0, a prioridade mais baixa, para 9, a prioridade mais alta	Uma mensagem enviada para o destino tem a prioridade especificada A prioridade padrão do produtor de mensagens, ou qualquer prioridade especificada na chamada Enviar, é ignorada. Se o destino for uma fila WebSphere MQ , o valor do atributo da fila <b>DefPriority</b> também será ignorado.
XMSC_PRIORITY_AS_APP	Uma mensagem enviada para o destino tem a prioridade especificada na chamada Enviar. Se a chamada Enviar especificar nenhuma prioridade, a prioridade padrão do produtor da mensagem será usada em seu lugar. Se o destino for uma fila WebSphere MQ , o valor do atributo da fila <b>DefPriority</b> será ignorado.
XMSC_PRIORITY_AS_DEST	Se o destino for uma fila WebSphere MQ , uma mensagem colocada na fila terá a prioridade especificada pelo valor do atributo da fila <b>DefPriority</b> . A prioridade padrão do produtor de mensagens, ou qualquer prioridade especificada na chamada Enviar, é ignorada.  Se o destino não for uma fila WebSphere MQ , o significado será o mesmo que XMSC_PRIORITY_AS_APP.

O valor-padrão é XMSC\_PRIORITY\_AS\_APP.

WebSphere MQ Transporte em Tempo Real e WebSphere MQ Multicast Transport não tomam nenhuma ação com base na prioridade de uma mensagem.

***XMSC\_PROVIDER\_NAME*****Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionMetaData

O provedor do cliente XMS . Essa propriedade é somente leitura.

***XMSC\_RTT\_BROKER\_PING\_INTERVAL*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O intervalo de tempo, em milissegundos, após o qual o XMS .NET verifica a conexão com um servidor do sistema de mensagens Real Time para detectar qualquer atividade. Se nenhuma atividade for detectada, o cliente iniciará um ping; a conexão será fechada se nenhuma resposta for detectada para o ping

O valor-padrão da propriedade é 30000.

## ***XMSC\_RTT\_CONNECTION\_PROTOCOL***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O protocolo de comunicações usado para uma conexão em tempo real com um broker.

O valor da propriedade deve ser `XMSC_RTT_CP_TCP`, o que significa uma conexão em tempo real com um broker sobre TCP/IP. O valor padrão é `XMSC_RTT_CP_TCP`.

## ***XMSC\_RTT\_HOST\_NAME***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome do host ou o endereço IP do sistema no qual um broker é executado.

Essa propriedade é usada com a propriedade `XMSC_RTT_PORT` para identificar o broker

Por padrão, a propriedade não é configurada

## ***XMSC\_RTT\_LOCAL\_ADDRESS***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome do host ou o endereço IP da interface de rede local a ser usada para uma conexão em tempo real com um broker.

Esta propriedade é útil apenas se o sistema no qual o aplicativo está em execução tiver duas ou mais interfaces de rede e você precisar ser capaz de especificar qual interface deve ser usada para uma conexão em tempo real. Se o sistema tiver apenas uma interface de rede, somente essa interface poderá ser usada. Se o sistema tiver duas ou mais interfaces de rede, e a propriedade não estiver configurada, a interface será selecionada aleatoriamente.

Por padrão, a propriedade não é configurada

## ***XMSC\_RTT\_MULTICAST***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory e Destino

**Nome usado em um URI:**

multicast

A configuração multicast para um connection factory ou destino. Apenas um destino que é um tópico pode ter essa propriedade

Um aplicativo usa esta propriedade para ativar multicast em associação com uma conexão em tempo real com um broker e, se multicast estiver ativado, para especificar a forma precisa na qual multicast é usado para entregar mensagens do broker para um consumidor de mensagem.. A propriedade não tem efeito sobre como um produtor de mensagem envia mensagens para o broker

Os valores válidos da propriedade são os seguintes:

**Valor válido**

XMSC\_RTT\_MULTICAST\_DISABLED

XMSC\_RTT\_MULTICAST\_ASCF

XMSC\_RTT\_MULTICAST\_ENABLED

XMSC\_RTT\_MULTICAST\_RELIABLE

XMSC\_RTT\_MULTICAST\_NOT\_RELIABLE

**Significado**

As mensagens não são entregues a um consumidor de mensagens usando WebSphere MQ Multicast Transport. Esse valor é o valor padrão para um objeto ConnectionFactory ..

As mensagens são entregues a um consumidor de mensagens de acordo com a configuração de multicast para o connection factory associado ao consumidor de mensagem. A configuração de multicast para o connection factory é observada no momento em que a conexão é criada. Esse valor é válido apenas para um objeto de Destino e é o valor padrão para um objeto de Destino..

Se o tópico for configurado para multicast no broker, as mensagens serão entregues a um consumidor de mensagem usando WebSphere MQ Multicast Transport. Uma qualidade de serviço confiável será usada, se o tópico for configurado para multicast confiável.

Se o tópico for configurado para multicast confiável no broker, as mensagens serão entregues a um consumidor de mensagens usando o WebSphere MQ Multicast Transport com uma qualidade de serviço confiável. Se o tópico não estiver configurado para multicast confiável, não será possível criar um consumidor de mensagens para o tópico.

Se o tópico for configurado para multicast no broker, as mensagens serão entregues a um consumidor de mensagem usando WebSphere MQ Multicast Transport. Uma qualidade de serviço confiável não será usada, mesmo se o tópico estiver configurado para multicast confiável.

***XMSC\_RTT\_PORT*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O número da porta na qual um broker atende às solicitações recebidas. No broker, deve-se configurar um nó de processamento de mensagens de Fluxo Real-timeInput ou Real-timeOptimized para atender nessa porta.

Essa propriedade é usada com a propriedade [XMSC\\_RTT\\_HOST\\_NAME](#) para identificar o broker.

O valor padrão da propriedade é [XMSC\\_RTT\\_DEFAULT\\_PORT](#) ou 1506.

***XMSC\_TIME\_TO\_LIVE*****Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

### Nome usado em um URI:

expiração (para um destino WebSphere MQ )

timeToLive (para um destino do provedor de sistemas de mensagens padrão do WebSphere )

O tempo de vida para mensagens enviadas para o destino.

Os valores válidos da propriedade são os seguintes:

Valor válido	Significado
0	Uma mensagem enviada ao destino nunca expira.
Um inteiro positivo	Uma mensagem enviada para o destino tem o tempo especificado para viver em milissegundos O tempo de vida padrão do produtor de mensagem, ou qualquer tempo de vida especificado na chamada Enviar, é ignorado.
XMSC_TIME_TO_LIVE_AS_APP	Uma mensagem enviada para o destino tem o tempo de vida especificado na chamada Enviar. Se a chamada de Envio especificar nenhum tempo de vida, o tempo de vida padrão do produtor da mensagem será usado em seu lugar

O valor padrão é XMSC\_TIME\_TO\_LIVE\_AS\_APP.

### ***XMSC\_USERID***

#### **Tipo de dado:**

Sequência

#### **Propriedade de:**

ConnectionFactory

Um identificador de usuário que pode ser usado para autenticar o aplicativo quando ele tenta se conectar a um servidor de mensagens. O identificador de usuário é usado com a propriedade XMSC\_PASSWORD .

Por padrão, a propriedade não é configurada

 Se você estiver se conectando ao IBM MQ for Multiplatformse configurar a propriedade XMSC\_USERID do connection factory, ele deverá corresponder ao **userid** do usuário conectado. Se você não configurar essas propriedades, o gerenciador de filas usará o **userid** do usuário conectado por padrão. Se você precisar de autenticação de nível de conexão adicional de usuários individuais, será possível gravar uma saída de autenticação de cliente configurada em IBM MQ.

 Para autenticar o usuário ao conectar ao IBM MQ for z/OS , é necessário usar uma saída de segurança.

### ***XMSC\_VERSION***

#### **Tipo de dado:**

Sequência

#### **Propriedade de:**

ConnectionMetaData

O identificador de versão do cliente XMS . Essa propriedade é somente leitura.

### ***XMSC\_WMQ\_BROKER\_CONTROLQ***

#### **Tipo de dado:**

Sequência

#### **Propriedade de:**

ConnectionFactory

O nome da fila de controle usada por um broker.

O valor padrão da propriedade é SYSTEM.BROKER.CONTROL.QUEUE.

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

### ***XMSC\_WMQ\_BROKER\_PUBQ***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome da fila monitorada por um broker na qual os aplicativos enviam mensagens que publicam.

O valor padrão da propriedade é SYSTEM.BROKER.DEFAULT.STREAM.

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

### ***XMSC\_WMQ\_BROKER\_QMGR***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome do gerenciador de filas ao qual um broker está conectado.

Por padrão, a propriedade não é configurada

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

### ***XMSC\_WMQ\_BROKER\_SUBQ***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome da fila de assinantes para um consumidor de mensagens não durável.

O nome da fila de assinantes deve começar com os seguintes caracteres:

SYSTEM.JMS.ND.

Se você deseja que todos os consumidores de mensagens não duráveis compartilhem uma fila de assinantes, especifique o nome completo da fila compartilhada. Uma fila com o nome especificado deve existir antes que um aplicativo possa criar um consumidor de mensagens não duráveis

Se deseja que cada consumidor de mensagens não duráveis recupere mensagens de sua própria fila de assinantes exclusivos, especifique um nome de fila que termine com um asterisco (\*). Em seguida, quando um aplicativo cria um consumidor de mensagens não duráveis, o cliente XMS cria uma fila dinâmica para uso exclusivo pelo consumidor de mensagens.. O cliente XMS usa o valor da propriedade para configurar o conteúdo do campo **DynamicQName** no descritor de objeto usado para criar a fila dinâmica.

O valor padrão da propriedade é SYSTEM.JMS.ND.SUBSCRIBER.QUEUE, que significa que XMS usa a abordagem de fila compartilhada por padrão.

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

### ***XMSC\_WMQ\_BROKER\_VERSION***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory e Destino

**Nome usado em um URI:**

brokerVersion

O tipo de corretor usado pelo aplicativo para uma conexão ou para o destino. Apenas um destino que é um tópico pode ter essa propriedade

Os valores válidos da propriedade são os seguintes:

Valor válido	Significado
XMSC_WMQ_BROKER_V1	O aplicativo está usando um broker do WebSphere MQ Publicação/Assinatura  O aplicativo também poderá usar esse valor se você migrar do WebSphere MQ Publicação/Assinatura para WebSphere Message Broker , mas não tiver alterado o aplicativo.
XMSC_WMQ_BROKER_V2	O aplicativo está usando um broker de IBM Integration Bus
XMSC_WMQ_BROKER_UNSPECIFIED	Após o broker ser migrado, configure essa propriedade para que os cabeçalhos RFH2 não sejam mais usados. Após a migração, essa propriedade não é mais relevante.

O valor padrão para um connectionfactory é XMSC\_WMQ\_BROKER\_UNESPECIFICADO mas, por padrão, a propriedade não é configurada para um destino. A configuração da propriedade para um destino substitui qualquer valor especificado pela propriedade do connection factory.

***XMSC\_WMQ\_CCDTURL*****Tipo de dado:**

System.String

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: CCDTURL

Nome abreviado da ferramenta de administração JMS: CCDT

Um Localizador Uniforme de Recursos (URL) que identifica o nome e o local do arquivo que contém a tabela de definição de canal do cliente e também especifica como o arquivo pode ser acessado.

Por padrão, essa propriedade não está configurada.

***XMSC\_WMQ\_CCSID*****Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

**Nome usado em um URI:**

CCSID

O identificador (CCSID) do conjunto de caracteres codificados, ou página de códigos, em que as cadeias de dados de caracteres no corpo de uma mensagem estão quando o cliente XMS encaminha a mensagem para o destino. Se configurado para uma mensagem individual, a propriedade JMS\_IBM\_CHARACTER\_SET substituirá o CCSID especificado para o destino por esta propriedade

O valor-padrão da propriedade é 1208.

Essa propriedade é relevante apenas para mensagens enviadas para o destino, não para mensagens recebidas do destino

## ***XMSC\_WMQ\_CHANEXO***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: CHANNEL

Nome abreviado da ferramenta de administração JMS: CHAN

O nome do canal a ser usado para uma conexão.

Por padrão, a propriedade não é configurada

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de fila no modo de cliente

## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_OPTIONS***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: CLIENTRECONNECTOPTIONS

Nome abreviado da ferramenta de administração JMS: CROPT

Esta propriedade especifica as opções de reconexão do cliente para novas conexões criadas por este factory. Ele é encontrado no XMSC e é um dos seguintes:

- WMQ\_CLIENT\_RECONNECT\_AS\_DEF (padrão).. Use o valor especificado no arquivo `mqclient.ini`. Configure o valor usando a propriedade **DefRecon** dentro da sub-rotina Canais. Ele pode ser configurado para um de:
  1. Sim. Comporta-se como a opção WMQ\_CLIENT\_RECONNECT
  2. NÃO. default. Não especifica nenhuma opção de reconexão
  3. QMGR. Comporta-se como a opção WMQ\_CLIENT\_RECONNECT\_Q\_MGR
  4. Desativar. Comporta-se como a opção WMQ\_CLIENT\_RECONNECT\_DISABLED
- WMQ\_CLIENT\_RECONNECT. Reconecte a qualquer um dos gerenciadores de filas especificados na lista de nomes de conexões.
- WMQ\_CLIENT\_RECONNECT\_Q\_MGR. Reconecta ao mesmo gerenciador de filas ao qual ele está originalmente conectado. Ele retorna MQRC\_RECONNECT\_QMID\_MISMATCH se o gerenciador de filas ao qual ele tenta se conectar (especificado na lista de nomes de conexão) tiver um QMID diferente para o gerenciador de filas originalmente conectado.
- WMQ\_CLIENT\_RECONNECT\_DISABLED. A reconexão está desativada

## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: CLIENTRECONNECTTIMEOUT

Nome abreviado da ferramenta de administração JMS: CRT

A propriedade `XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT` é válida apenas para o cliente XMS .NET Gerenciado

Esta propriedade especifica a duração de tempo, em segundos, que uma conexão do cliente tenta reconectar.

Depois de tentar se reconectar por esse tempo, o cliente falhará com `MQRC_RECONNECT_FAILED`. A configuração padrão para esta propriedade é `XMSC.WMQ_CLIENT_RECONNECT_TIMEOUT_DEFAULT`.

O valor-padrão desta propriedade é 1800.

### ***XMSC\_WMQ\_CONNECTION\_MODE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O modo pelo qual um aplicativo se conecta a um gerenciador de filas.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
<code>XMSC_WMQ_CM_BINDINGS</code>	Uma conexão com um gerenciador de fila no modo de ligações, para obter um desempenho ideal Esse valor é o valor padrão para C/C + +.
<code>XMSC_WMQ_CM_CLIENT</code>	Uma conexão com um gerenciador de filas no modo cliente para assegurar uma pilha totalmente gerenciada. Esse valor é o valor padrão para .NET.
<code>XMSC_WMQ_CM_CLIENT_UNMANAGED</code> (apenas para .NET)	Uma conexão com um gerenciador de filas que força uma pilha de clientes não gerenciada

### **Conceitos relacionados**

Operações gerenciadas e não gerenciadas no .NET

O código gerenciado é executado exclusivamente dentro do ambiente Common Language Runtime do .NET e depende totalmente dos serviços fornecidos por esse tempo de execução. Um aplicativo é classificado como não gerenciado se qualquer parte do aplicativo executar ou chamar serviços fora do ambiente de tempo de execução de linguagem comum .NET .

### ***XMSC\_WMQ\_CONNECTION\_NAME\_LIST***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: `CONNECTIONNAMELIST`

Nome abreviado da ferramenta de administração JMS: `CNLIST`

Esta propriedade especifica os hosts aos quais o cliente tenta se reconectar depois que sua conexão é interrompida

A lista de nomes de conexão é uma lista separada por vírgula de pares de portas host / ip. A configuração padrão para essa propriedade é `WMQ_CONNECTION_NAME_LIST_DEFAULT`

Por exemplo, `127.0.0.1 (1414) , host2.example.com(1400)`

A configuração padrão desta propriedade é `localhost (1414)`.

## ***XMSC\_WMQ\_DUR\_SUBQ***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

Destino

O nome da fila de assinantes para um assinante durável que está recebendo mensagens do destino. Apenas um destino que é um tópico pode ter essa propriedade

O nome da fila de assinantes deve começar com os seguintes caracteres:

SYSTEM.JMS.D.

Se você deseja que todos os assinantes duráveis compartilhem uma fila de assinantes, especifique o nome completo da fila compartilhada. Uma fila com o nome especificado deve existir antes que um aplicativo possa criar um assinante durável

Se deseja que cada assinante durável recupere mensagens de sua própria fila de assinantes exclusivos, especifique um nome da fila que termine com um asterisco (\*). Então, quando um aplicativo cria um assinante durável, o cliente XMS cria uma fila dinâmica para uso exclusivo pelo assinante durável. O cliente XMS usa o valor da propriedade para configurar o conteúdo do campo **DynamicQName** no descritor de objeto usado para criar a fila dinâmica.

O valor padrão da propriedade é SYSTEM.JMS.D.SUBSCRIBER.QUEUE, que significa que XMS usa a abordagem de fila compartilhada por padrão.

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

## ***XMSC\_WMQ\_ENCODING***

### **Tipo de dado:**

System.Int32

### **Propriedade de:**

Destino

Como os dados numéricos no corpo de uma mensagem são representados quando o cliente XMS encaminha a mensagem para o destino. Se configurado para uma mensagem individual, a propriedade JMS\_IBM\_ENCODING substituirá a codificação especificada para o destino por essa propriedade. A propriedade especifica a representação de números inteiros binários, números inteiros decimais empacotados e números de vírgula flutuante..

Os valores válidos da propriedade são iguais aos valores que podem ser especificados no campo **Encoding** de um descritor de mensagens.

Um aplicativo pode usar as seguintes constantes nomeadas para configurar a propriedade:

<b>Constante nomeada</b>	<b>Significado</b>
MQENC_INTEGER_NORMAL	Codificação de número inteiro normal
MQENC_INTEGER_REVERSED	Codificação de número inteiro revertida
MQENC_DECIMAL_NORMAL	Codificação decimal compactada normal
MQENC_DECIMAL_REVERSED	Codificação decimal compactada revertida
MQENC_FLOAT_IEEE_NORMAL	Codificação de ponto flutuante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codificação de ponto flutuante IEEE revertida
MQENC_FLOAT_S390	Codificação de ponto flutuante da arquitetura do z/OS
MQENC_NATIVE	Codificação de máquina nativa

Para formar um valor para a propriedade, o aplicativo pode incluir três dessas constantes da seguinte forma:

- Uma constante cujo nome começa com MQENC\_INTEGER, para especificar a representação de inteiros binários
- Uma constante cujo nome começa com MQENC\_DECIMAL, para especificar a representação de números inteiros decimais compactados
- Uma constante cujo nome começa com MQENC\_FLOAT, para especificar a representação de números de ponto flutuante

Como alternativa, o aplicativo pode configurar a propriedade para MQENC\_NATIVE, cujo valor é dependente de ambiente.

O valor padrão da propriedade é MQENC\_NATIVE.

Essa propriedade é relevante apenas para mensagens enviadas para o destino, não para mensagens recebidas do destino

### ***XMSC\_WMQ\_FAIL\_IF QUIESCE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory e Destino

**Nome usado em um URI:**

failIfQuiesce

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: FAILIFQUIESCE

Nome abreviado da ferramenta de administração JMS: FIQ

Se as chamadas para determinados métodos falharão se o gerenciador de filas ao qual o aplicativo está conectado estiver em um estado quiesce.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
XMSC_WMQ_FIQ_YES	Chamadas para determinados métodos falharão se o gerenciador de filas estiver em um estado quiesce. Quando o aplicativo detectar que o gerenciador de filas está em quiesce, o aplicativo poderá concluir sua tarefa imediata e fechar a conexão, permitindo que o gerenciador de filas pare.
XMSC_WMQ_FIQ_NO	Nenhuma chamada de método falha porque o gerenciador de fila está em um estado de quiesce. Se você especificar esse valor, o aplicativo não poderá detectar que o gerenciador de filas está quiesce. O aplicativo pode continuar a executar operações no gerenciador de fila e, portanto, evitar que o gerenciador de filas pare.

O valor padrão para um connection factory é XMSC\_WMQ\_FIQ\_YES mas, por padrão, a propriedade não é configurada para um destino. A configuração da propriedade para um destino substitui qualquer valor especificado pela propriedade do connection factory.

### ***XMSC\_WMQ\_MESSAGE\_BODY***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

Essa propriedade determina se um aplicativo XMS processa o MQRFH2 de uma mensagem IBM WebSphere MQ como parte da carga útil da mensagem (ou seja, como parte do corpo da mensagem).

**Nota:** Ao enviar mensagens para um destino, a propriedade XMSC\_WMQ\_MESSAGE\_BODY substitui a propriedade de destino XMSC\_WMQ\_TARGET\_CLIENT existente do XMS

Os valores válidos para essa propriedade são:

#### **XMSC\_WMQ\_MESSAGE\_BODY\_JMS**

**Receber:** O tipo de mensagem e o corpo do XMS de entrada são determinados pelo conteúdo do MQRFH2 (se presente) ou do MQMD (se não houver MQRFH2) na mensagem recebida IBM WebSphere MQ .

**Enviar:** O corpo da mensagem XMS de saída contém um cabeçalho MQRFH2 pré-anexado e gerado automaticamente com base nas propriedades da mensagem e nos campos do cabeçalho do XMS

#### **XMSC\_WMQ\_MESSAGE\_BODY\_MQ**

**Receber:** O tipo de mensagem XMS de entrada é sempre ByteMessage, independentemente do conteúdo da mensagem recebida do IBM WebSphere MQ ou do campo de formato do MQMD recebido. O corpo da mensagem XMS é os dados da mensagem inalterados retornados pela chamada API do provedor de sistemas de mensagens subjacente. O conjunto de caracteres e a codificação dos dados no corpo da mensagem são determinados pelos campos CodedCharSetId e Codificação do MQMD. O formato dos dados no corpo da mensagem é determinado pelo campo Formato do MQMD.

**Enviar:** o corpo da mensagem XMS de saída contém a carga útil do aplicativo no estado em que se encontra e nenhum cabeçalho IBM WebSphere MQ gerado automaticamente é incluído no corpo.

#### **XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED**

**Recebimento:** O cliente XMS determina um valor adequado para essa propriedade.. No caminho de recebimento, esse valor é o valor da propriedade WMQ\_MESSAGE\_BODY\_JMS.

**Enviar:** O cliente XMS determina um valor adequado para essa propriedade. No caminho de envio, esse valor é o valor da propriedade XMSC\_WMQ\_TARGET\_CLIENT.

Por padrão, essa propriedade é configurada como XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED

#### **XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT**

##### **Tipo de dado:**

System.Int32

##### **Propriedade de:**

Destino

Determina qual nível de contexto da mensagem deve ser configurado pelo aplicativo XMS . O aplicativo deve estar em execução com autoridade de contexto apropriado para esta propriedade entrar em vigor.

Os valores válidos para essa propriedade são:

#### **XMSC\_WMQ\_MDCTX\_DEFAULT**

Para mensagens de saída, a chamada da API MQOPEN e a estrutura MQPMO não especificam opções de contexto de mensagem explícitas

#### **XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT**

A chamada da API MQOPEN especifica a opção de contexto da mensagem MQOO\_SET\_IDENTITY\_CONTEXT e a estrutura MQPMO especifica MQPMO\_SET\_IDENTITY\_CONTEXT

#### **XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT**

A chamada da API MQOPEN especifica a opção de contexto da mensagem MQOO\_SET\_ALL\_CONTEXT e a estrutura MQPMO especifica MQPMO\_SET\_ALL\_CONTEXT

Por padrão, essa propriedade é configurada para XMSC\_WMQ\_MDCTX\_DEFAULT

**Nota:** Essa propriedade não é relevante quando um aplicativo se conecta ao Integration Busdo sistema.

As propriedades a seguir requerem que a propriedade XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT seja configurada como o valor da propriedade XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT ou o valor da propriedade XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT ao enviar uma mensagem para que tenha efeito desejado:

- JMS\_IBM\_MQMD\_UserIdentifier

- JMS\_IBM\_MQMD\_AccountingToken
- JMS\_IBM\_MQMD\_ApplIdentityData

As propriedades a seguir requerem que a propriedade XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT seja configurada para o valor da propriedade XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT ao enviar uma mensagem para que tenha efeito desejado:

- 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 dado:**

System.Int32

**Propriedade de:**

Destino

Essa propriedade determina se um aplicativo XMS pode extrair os valores de campos MQMD ou não. Os valores válidos para essa propriedade são:

#### **XMSC\_WMQ\_READ\_ENABLED\_NO**

Ao enviar mensagens, as propriedades JMS\_IBM\_MQMD\* em uma mensagem enviada não são atualizadas para refletir os valores de campos atualizados no MQMD.

Ao receber mensagens, nenhuma das propriedades JMS\_IBM\_MQMD\* está disponível em uma mensagem recebida, mesmo que algumas ou todas elas sejam configuradas pelo emissor.

#### **XMSC\_WMQ\_READ\_ENABLED\_YES**

Ao enviar mensagens, todas as propriedades JMS\_IBM\_MQMD\* em uma mensagem enviada são atualizadas para refletir os valores de campo atualizados no MQMD, incluindo as propriedades que o emissor não configurou explicitamente...

Ao receber mensagens, todas as propriedades JMS\_IBM\_MQMD\* estão disponíveis em uma mensagem recebida, incluindo aquelas propriedades que o emissor não configurou explicitamente.

Por padrão, essa propriedade é configurada como XMSC\_WMQ\_READ\_ENABLED\_NO

### ***XMSC\_WMQ\_MQMD\_WRITE\_ENABLED***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

Esta propriedade determina se um aplicativo XMS pode ou não os valores de campos MQMD.

Os valores válidos para essa propriedade são:

#### **XMSC\_WMQ\_WRITE\_ENABLED\_NO**

Todas as propriedades JMS\_IBM\_MQMD\* são ignoradas e seus valores não são copiados para a estrutura MQMD subjacente.

#### **XMSC\_WMQ\_WRITE\_ENABLED\_YES**

As propriedades JMS\_IBM\_MQMD\* são processadas. Seus valores serão copiados para a estrutura do MQMD subjacente.

Por padrão, essa propriedade é configurada como XMSC\_WMQ\_WRITE\_ENABLED\_NO

## ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

Essa propriedade determina se os produtores de mensagens têm permissão para usar as postagens assíncronas para enviar mensagens para esse destino.

Os valores válidos para essa propriedade são:

### ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST***

Determine se as colocações assíncronas são permitidas consultando a definição de fila ou tópico.

### ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_Q\_DEF***

Determine se as entradas assíncronas são permitidas consultando a definição de fila.

### ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_TOPIC\_DEF***

Determine se as colocações assíncronas são permitidas referindo-se à definição de tópico

### ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_DISABLED***

Puts assíncronos não são permitidos.

### ***XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_ENABLED***

As colocações assíncronas são permitidas

Por padrão, essa propriedade é configurada como *XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST*

**Nota:** Esta propriedade não é relevante quando um aplicativo está se conectando ao Integration Busdo sistema

## ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

Essa propriedade determina se os consumidores de mensagens e os navegadores de fila têm permissão para usar leitura antecipada para obter mensagens não persistentes, não transacionais desse destino em um buffer interno antes de recebê-las.

Os valores válidos para essa propriedade são:

### ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF***

Determine se a leitura antecipada é permitida referindo-se à definição de fila

### ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TÓPICO\_DEF***

Determine se a leitura antecipada é permitida referindo-se à definição de tópico

### ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST***

Determine se a leitura antecipada é permitida consultando a definição de fila ou tópico.

### ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED***

A leitura antecipada não é permitida ao consumir ou procurar mensagens.

### ***XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_ENABLED***

A leitura antecipada é permitida

Por padrão, essa propriedade é configurada como *XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST*

## ***XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

Para mensagens que estão sendo entregues em um listener de mensagem assíncrona, essa propriedade determina o que acontece com as mensagens no buffer de leitura antecipada interno quando o consumidor de mensagens é fechado.

Essa propriedade é aplicável ao especificar opções de fila de fechamento ao consumir mensagens de um destino e não aplicável ao enviar mensagens para um destino.

Essa propriedade é ignorada para Navegadores de Fila, pois durante a procura as mensagens ainda estão disponíveis nas filas.

Os valores válidos para essa propriedade são:

### ***XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT***

Apenas a chamada do listener de mensagem atual é concluída antes de retornar, potencialmente deixando mensagens no buffer de leitura antecipada interno, que são, então, descartados

### ***XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_ALL***

Todas as mensagens no buffer de leitura antecipada interno são entregues para o listener de mensagens do aplicativo antes de retornar.

Por padrão, essa propriedade é configurada como `XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT`

**Nota:**

- **Terminação anormal do aplicativo**

Todas as mensagens no buffer de leitura antecipada são perdidas quando um aplicativo XMS termina abruptamente.

- **Implicações nas transações**

A leitura antecipada é desativada quando os aplicativos usam transação. Portanto, o aplicativo não está vendo nenhuma diferença no comportamento quando eles usam sessões transacionados

- **Implicações dos modos de Confirmação de sessão**

A leitura antecipada é ativada em uma sessão não transacionada quando os modos de confirmação são `XMSC_AUTO_RECONHEÇO` ou `XMSC_DUPS_OK_RECONHEÇO`. A leitura antecipada será desativada se o modo de confirmação de sessão for `XMSC_CLIENT_RECONHEÇO`, independentemente de sessões transacionadas ou não transacionados

- **Implicações em Navegadores de Fila e Seletores de Navegador de Fila**

Os Navegadores de Fila e os Seletores de Navegador de Filas, usados em aplicativos XMS, obtêm a vantagem de desempenho da leitura antecipada. Fechar o Queue Browser não compromete o desempenho, pois a mensagem ainda está disponível na fila para operações adicionais. Não há qualquer outra implicação em navegadores de filas e seletores de navegadores de filas além dos benefícios de desempenho de leitura antecipada.

## ***XMSC\_WMQ\_HOST\_NAME***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: HOSTNAME

Nome abreviado da ferramenta de administração JMS: HOST

O nome do host ou o endereço IP do sistema no qual um gerenciador de filas é executado.

Essa propriedade é usada apenas quando um aplicativo se conecta a um gerenciador de filas em modo cliente. A propriedade é utilizada com a propriedade `XMSC_WMQ_PORT` para identificar o gerenciador de filas.

O valor padrão da propriedade é `localhost..`

## ***XMSC\_WMQ\_LOCAL\_ADDRESS***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

ConnectionFactory

### **Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: LOCALADDRESS

Nome abreviado da ferramenta de administração JMS: LA

Para uma conexão com um gerenciador de filas, essa propriedade especifica a interface de rede local a ser usada, a porta local ou o intervalo de portas locais a serem usadas ou ambos.

O valor da propriedade é uma sequência com o seguinte formato:

`[host_name] [(low_port) [,high_port]]`

Os significados das variáveis são os seguintes:

### ***host\_name***

O nome do host ou o endereço IP da interface de rede local a ser usado para a conexão

Fornecer essas informações é necessário apenas se o sistema no qual o aplicativo está em execução tiver duas ou mais interfaces de rede e você precisar ser capaz de especificar qual interface deve ser usada para a conexão.. Se o sistema tiver apenas uma interface de rede, somente essa interface poderá ser usada. Se o sistema tiver duas ou mais interfaces de rede e você não especificar qual interface deve ser usada, a interface será selecionada aleatoriamente

### ***low\_port***

O número da porta local a ser usada para a conexão

Se *high\_port* também for especificado, *low\_port* será interpretado como o número da porta mais baixo em um intervalo de números de portas

### ***high\_port***

O número da porta mais alto em um intervalo de números de porta. Uma das portas no intervalo especificado deve ser usada para a conexão..

O comprimento máximo da sequência é de 48 caracteres.

Aqui estão alguns exemplos de valores válidos da propriedade:

```
JÚPITER
9.20.4.98
JUPITER (1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)
```

Por padrão, a propriedade não é configurada

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de fila no modo de cliente

## ***XMSC\_WMQ\_MESSAGE\_SELECTION***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

Determina se a seleção de mensagem é feita pelo cliente XMS ou pelo broker

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
XMSC_WMQ_MSEL_CLIENT	A seleção de mensagens é feita pelo cliente do XMS
XMSC_WMQ_MSEL_BROKER	A seleção de mensagens é feita pelo broker

O valor padrão é XMSC\_WMQ\_MSEL\_CLIENT.

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura. A seleção de mensagens pelo broker não será suportada se a propriedade XMSC\_WMQ\_BROKER\_VERSION estiver configurada como XMSC\_WMQ\_BROKER\_V1.

## ***XMSC\_WMQ\_MSG\_BATCH\_SIZE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O número máximo de mensagens a serem recuperadas de uma fila em um lote ao usar a entrega de mensagem assíncrona.

Quando um aplicativo está usando a entrega de mensagem assíncrona, sob determinadas condições, o cliente XMS recupera um lote de mensagens de uma fila antes de encaminhar cada mensagem individualmente para o aplicativo. Esta propriedade especifica o número máximo de mensagens que podem estar no lote.

O valor da propriedade é um inteiro positivo e o valor padrão é 10. Considere configurar a propriedade para um valor diferente somente se você tiver um problema de desempenho específico que precisa ser abordado.

Se um aplicativo for conectado a um gerenciador de filas por meio de uma rede, aumentar o valor dessa propriedade pode reduzir sobrecargas de rede e tempos de resposta, mas aumentar a quantidade de memória necessária para armazenar as mensagens no sistema do cliente. Por outro lado, reduzir o valor dessa propriedade pode aumentar as sobrecargas de rede e os tempos de resposta, mas reduzir a quantidade de memória necessária para armazenar as mensagens..

## ***XMSC\_WMQ\_POLLING\_INTERVAL***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

Se cada listener de mensagem dentro de uma sessão não tiver mensagens adequadas em sua fila, este valor será o intervalo máximo, em milissegundos, que decorrerá antes que cada listener da mensagem tente novamente obter uma mensagem de sua fila.

Se ocorrer com frequência o fato de nenhuma mensagem adequada estar disponível para qualquer um dos listeners da mensagem em uma sessão, considere aumentar o valor desta propriedade.

O valor da propriedade é um número inteiro positivo. O valor-padrão é 5000.

## ***XMSC\_WMQ\_PORT***

### **Tipo de dado:**

System.Int32

### **Propriedade de:**

ConnectionFactory

### **Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: PORT

Nome abreviado da ferramenta de administração JMS: PORT

O número da porta na qual um gerenciador de filas atende às solicitações recebidas.

Essa propriedade é usada apenas quando um aplicativo se conecta a um gerenciador de filas em modo cliente. A propriedade é usada com a propriedade XMSC\_WMQ\_HOST\_NAME para identificar o gerenciador de filas.

O valor padrão da propriedade é XMSC\_WMQ\_DEFAULT\_CLIENT\_PORT ou 1414.

## ***XMSC\_WMQ\_PROVIDER\_VERSION***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

ConnectionFactory

A versão, liberação, nível de modificação e fix pack do gerenciador de filas ao qual o aplicativo pretende se conectar. Os valores válidos para essa propriedade são:

- Não especificado

Ou uma sequência em um dos seguintes formatos

- V.R.M.F
- V.R.M
- V.R
- V

em que V, R, M e F são valores inteiros maiores ou iguais a zero.

Um valor de 7 ou superior indica que esta versão é destinada como um IBM WebSphere MQ 7.0 ConnectionFactory para conexões com um gerenciador de filas do IBM WebSphere MQ 7.0 .. Um valor anterior a 7 (por exemplo "6.0.2.0"), indica que ele é destinado para uso com gerenciadores de filas anteriores à Versão 7.0. O valor padrão, não especificado, permite conexões com qualquer nível do gerenciador de filas, determinando as propriedades e a funcionalidade aplicáveis disponíveis com base nos recursos do gerenciador de filas...

Por padrão, essa propriedade é configurada como "não especificada"..

### **Nota:**

- Nenhum compartilhamento de soquete ocorrerá se XMSC\_WMQ\_PROVIDER\_VERSION estiver configurado como 6 2.
- A conexão falhará se XMSC\_WMQ\_PROVIDER\_VERSION estiver configurado como 7 e no servidor SHARECNV para o canal estiver configurado como 0
- Os recursos específicos do IBM WebSphere MQ 7.0 serão desativados se XMSC\_WMQ\_PROVIDER\_VERSION estiver configurado como UNSPECIFIED e SHARECNV estiver configurado como 0..

A versão do IBM WebSphere MQ Client também desempenha uma função principal se um aplicativo cliente XMS pode usar recursos específicos do IBM WebSphere MQ 7.0 . A tabela a seguir descreve o comportamento..

**Nota:** Uma propriedade de sistema XMSC\_WMQ\_OVERRIDEPROVIDERVERSION substitui a propriedade XMSC\_WMQ\_PROVIDER\_VERSION.. Esta propriedade pode ser usada se você não conseguir alterar a configuração do connection factory

*Tabela 33. cliente XMS -Capacidade de usar recursos específicos do IBM WebSphere MQ 7.0 .*

#	XMSC_WMQ_PROVIDER_VERSION	Versão do cliente IBM WebSphere MQ	Recursos do IBM WebSphere MQ 7.0
1	não especificado	7	ATIVADA
2	não especificado	6	Desativado
3	7	7	ATIVADA
4	7	6	Exceção
5	6	6	Desativado
6	6	7	Desativado

### **XMSC\_WMQ\_PUB\_ACK\_INTERVAL**

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O número de mensagens publicadas por um publicador antes do cliente XMS solicitar uma confirmação do broker.

Se você diminuir o valor dessa propriedade, o cliente solicita confirmações com mais frequência e, portanto, o desempenho do publicador diminui. Se aumentar o valor, o cliente terá mais tempo para emitir uma exceção se o intermediário falhar.

O valor da propriedade é um número inteiro positivo. O valor padrão é 25.

### **XMSC\_WMQ\_QMGR\_CCSID**

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O identificador (CCSID) do conjunto de caracteres codificados, ou página de código, no qual os campos de dados de caracteres definidos na Message Queue Interface (MQI) são trocados entre o cliente XMS e o cliente WebSphere MQ . Essa propriedade não se aplica às cadeias dos dados de caracteres nos corpos das mensagens.

Quando o aplicativo um XMS se conecta a um gerenciador de fila no modo cliente, o cliente XMS vincula ao cliente WebSphere MQ . As informações trocadas entre os dois clientes contêm campos de dados de caracteres definidos no MQI. Sob circunstâncias normais, o cliente WebSphere MQ assume que esses campos estão na página de códigos do sistema no qual os clientes estão em execução Se o cliente XMS fornecer e esperar receber esses campos em uma página de código diferente, deve-se configurar essa propriedade para informar o cliente WebSphere MQ

Quando o cliente WebSphere MQ encaminha esses campos de dados de caractere para o gerenciador de filas, os dados neles devem ser convertidos, se necessário, na página de códigos usada pelo gerenciador de fila Da mesma forma, quando o cliente WebSphere MQ recebe esses campos do gerenciador de filas, os dados neles devem ser convertidos, se necessário, para a página de código na qual o cliente XMS espera receber os dados O cliente WebSphere MQ usa essa propriedade para executar essas conversão de dados.

Por padrão, a propriedade não é configurada

A configuração dessa propriedade é equivalente à configuração da variável de ambiente MQCCSID para um cliente WebSphere MQ que suporta aplicativos clientes nativos WebSphere MQ . Para obter informações adicionais sobre esta variável de ambiente, consulte *WebSphere MQ Clientes*.

### ***XMSC\_WMQ\_QUEUE\_MANAGER***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: QMANAGER

Nome abreviado da ferramenta de administração JMS: QMGR

O nome do gerenciador de filas para conexão.

Por padrão, a propriedade não é configurada

### ***XMSC\_WMQ\_RECEIVE\_CCSID***

A propriedade de destino que configura o destino CCSID para a conversão de mensagens do gerenciador de filas. O valor é ignorado a menos que XMSC\_WMQ\_RECEIVE\_CONVERSION seja configurado como WMQ\_RECEIVE\_CONVERSION\_QMGR.

**Tipo de dado:**

Integer

**Valor:**

Qualquer número inteiro positivo.

O valor padrão é 1208.

Especificar um valor GMO\_CONVERT em uma mensagem é opcional. Se um valor GMO\_CONVERT for especificado, a conversão ocorrerá de acordo com o valor especificado.

### ***XMSC\_WMQ\_RECEIVE\_CONVERSION***

A propriedade de destino que determina se a conversão de dados será executada pelo gerenciador de filas.

**Tipo de dado:**

Integer

**Valores:**

XMSC\_WMQ\_RECEIVE\_CONVERSION\_CLIENT\_MSG (DEFAULT): Execute a conversão de dados somente no cliente XMS A conversão é sempre feita usando a página de códigos 1208

XMSC\_WMQ\_RECEIVE\_CONVERSION\_QMGR: Executar conversão de dados no gerenciador de filas antes de enviar uma mensagem ao cliente XMS .

### ***XMSC\_WMQ\_RECEIVE\_EXIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Identifica uma saída de recebimento do canal para ser executada.

O valor da propriedade é uma sequência que identifica uma saída de recebimento do canal e possui o seguinte formato:

**libraryName**(entryPointNome)

em que,

- **libraryName** é o caminho completo da saída gerenciada .dll
- **entryPointName** é o nome da classe qualificado pelo namespace.

Por exemplo, C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)

Por padrão, a propriedade não é configurada

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado Além disso, apenas as saídas gerenciadas são suportadas

### ***XMSC\_WMQ\_RECEIVE\_EXIT\_INIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Os dados do usuário que são transmitidos para uma saída de recebimento de canal quando ela é chamada.

O valor da propriedade é uma sequência. Por padrão, a propriedade não é configurada

Essa propriedade é relevante apenas quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado e a propriedade [“XMSC\\_WMQ\\_RECEIVE\\_EXIT”](#) na página 232 é configurada

### ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Essa propriedade é usada para obter o nome do gerenciador de filas ao qual ele está conectado

Quando usado com uma CCDT (Client Channel Definition Table), esse nome pode ser diferente do nome do gerenciador de filas especificado na Connection Factory.

### ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER\_ID***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Essa propriedade é preenchida com o ID do Gerenciador de Filas após a conexão.

### ***XMSC\_WMQ\_SECURITY\_EXIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Identifica uma saída de segurança do canal.

O valor da propriedade é uma sequência que identifica uma saída de segurança do canal e possui o seguinte formato:

**libraryName**(entryPointNome)

em que,

- **libraryName** é o caminho completo da saída gerenciada .dll
- **entryPointName** é o nome da classe qualificado pelo namespace.

Por exemplo, C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

O comprimento máximo da cadeia é de 128 caracteres.

Por padrão, a propriedade não é configurada

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado Além disso, apenas as saídas gerenciadas são suportadas

### ***XMSC\_WMQ\_SECURITY\_EXIT\_INIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Os dados do usuário que são transmitidos para uma saída de segurança do canal quando ela é chamada.

O comprimento máximo da sequência de dados do usuário é de 32 caracteres.

Por padrão, a propriedade não é configurada

Essa propriedade é relevante apenas quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado e a propriedade [“XMSC\\_WMQ\\_SECURITY\\_EXIT”](#) na página 233 é configurada

### ***XMSC\_WMQ\_SEND\_EXIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Identifica uma saída de envio de canal.

O valor da propriedade é uma sequência. Uma saída de envio de canal tem o seguinte formato:

**libraryName**(entryPointNome)

em que,

- **libraryName** é o caminho completo da saída gerenciada .dll
- **entryPointNome** é o nome da classe qualificado pelo namespace.

Por exemplo, C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

Por padrão, a propriedade não é configurada

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado Além disso, apenas as saídas gerenciadas são suportadas

### ***XMSC\_WMQ\_SEND\_EXIT\_INIT***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Os dados do usuário que são transmitidos para as saídas de envio do canal quando são chamadas.

O valor da propriedade é uma sequência de um ou mais itens de dados do usuário separados por vírgula.

Por padrão, a propriedade não é configurada

As regras para especificar dados do usuário que são transmitidos para uma sequência de saídas de envio de canal são as mesmas que as regras para especificar dados do usuário que são transmitidos para uma sequência de saídas de recebimento de canal.. Portanto, para obter as regras, consulte [“XMSC\\_WMQ\\_RECEIVE\\_EXIT\\_INIT”](#) na página 233

Essa propriedade é relevante apenas quando um aplicativo se conecta a um gerenciador de filas no modo de cliente gerenciado e a propriedade “XMSC\_WMQ\_SEND\_EXIT” na página 234 é configurada

### ***XMSC\_WMQ\_SEND\_CHECK\_COUNT***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O número de chamadas de envio a serem permitidas entre a verificação de erros de postagem assíncrona, dentro de uma única sessão XMS não transacionada.

Por padrão, esta propriedade é configurada como 0

### ***XMSC\_WMQ\_SHARE\_CONV\_ALLOWED***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

**Objetos aplicáveis:**

Nome longo da ferramenta de administração JMS: SHARECONVALLOWED

Nome abreviado da ferramenta de administração JMS: SCALD

Se uma conexão do cliente pode compartilhar seu soquete com outras conexões XMS de nível superior do mesmo processo para o mesmo gerenciador de filas, se as definições de canais corresponderem. Essa propriedade é fornecida para permitir o isolamento completo de Conexões em soquetes separados, se necessário para desenvolvimento de aplicativos, manutenção ou razões operacionais. A configuração dessa propriedade apenas indica para XMS para tornar o soquete subjacente compartilhado. Ele não indica quantas conexões compartilham um único soquete. O número de conexões compartilhando um soquete é determinado pelo valor SHARECNV que é negociado entre o WebSphere MQ Client e o WebSphere MQ Server.

Um aplicativo pode configurar as seguintes constantes nomeadas para configurar a propriedade:

- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_FALSE-As conexões não compartilham um soquete
- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_TRUE-As conexões compartilham um soquete

Por padrão, a propriedade é configurada como XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_ENABLED

Essa propriedade é relevante somente quando um aplicativo se conecta a um gerenciador de fila no modo de cliente

### ***XMSC\_WMQ\_SSL\_CERT\_STORES***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Os locais dos servidores que retêm as listas de revogação de certificado (CRLs) a serem usadas em uma conexão SSL com um gerenciador de filas.

O valor da propriedade é uma lista de uma ou mais URLs separadas por vírgulas. Cada URL tem o seguinte formato:

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

Esse formato é compatível com, mas estendido do, formato MQJMS básico.

É válido ter um `serveraddressvazio`. Neste caso, XMS assume que o valor é a sequência "localhost".

Uma lista de exemplo é:

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

Por padrão, a propriedade não é configurada

#### Informações relacionadas

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

### ***XMSC\_WMQ\_SSL\_CIPHER\_SPEC***

#### Tipo de dado:

Sequência

#### Propriedade de:

ConnectionFactory

O nome da CipherSpec a ser usada em uma conexão segura com um gerenciador de filas.

As especificações de código que podem ser usadas com o suporte TLS do IBM WebSphere MQ são listadas na tabela a seguir. Ao exigir um certificado pessoal, você especifica um tamanho de chave para o par de chaves público e particular. O tamanho de chave que é usado durante o handshake SSL é o tamanho armazenado no certificado, a menos que ele seja determinado pelo CipherSpec, conforme indicado na tabela. Por padrão, essa propriedade não está configurada.

Nome do CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de criptografia	Bits de Criptografia	FIPS <sup>1</sup>	Conjunto B de 128 bits	Conjunto B de 192 bits
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	Padrão de Criptografia Avançado	128	Sim	Não	Não
TLS_RSA_WITH_AES_256_CBC_SHA <sup>2</sup>	TLS 1.0	SHA-1	Padrão de Criptografia Avançado	256	Sim	Não	Não
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	Padrão de Criptografia de Dados	56	Não	Não	Não
TLS_RSA_WITH_3DES_EDE_CBC_SHA <sup>4</sup>	TLS 1.0	SHA-1	3DES	168	Sim	Não	Não

Nome do CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de criptografia	Bits de Criptografia	FIPS <sup>1</sup>	Conjunto B de 128 bits	Conjunto B de 192 bits
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Não	Não
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	Padrão de Criptografia Avançado	256	Sim	Não	Não
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Não	Não
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	256	Sim	Não	Não
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Não	Não	Não
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Sim	Não	Não
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Não	Não	Não
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Sim	Não	Não
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Não	Não
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	Padrão de Criptografia Avançado	256	Sim	Não	Não

Nome do CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de criptografia	Bits de Criptografia	FIPS <sup>1</sup>	Conjunto B de 128 bits	Conjunto B de 192 bits
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Não	Não
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	Padrão de Criptografia Avançado	256	Sim	Não	Não
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Sim	Não
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	Padrão de Criptografia Avançado	256	Sim	Não	Sim
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	Padrão de Criptografia Avançado	128	Sim	Não	Não
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	Padrão de Criptografia Avançado	256	Sim	Não	Não
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	Nenhum	0	Não	Não	Não
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	Nenhum	0	Não	Não	Não
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	Nenhum	0	Não	Não	Não
TLS_RSA_WITH_NULL_NULL	TLS 1.2	Nenhum	Nenhum	0	Não	Não	Não
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Não	Não	Não

Nome do CipherSpec	Protocolo utilizado	Algoritmo hash	Algoritmo de criptografia	Bits de Criptografia	FIPS <sup>1</sup>	Conjunto B de 128 bits	Conjunto B de 192 bits
--------------------	---------------------	----------------	---------------------------	----------------------	-------------------	------------------------	------------------------

**Notes:**

1. Especifica se o CipherSpec está em conformidade com o Federal Information Processing Standards (FIPS) 140-2. Para obter uma explicação do FIPS e informações sobre como configurar a operação compatível com o WebSphere MQ para FIPS 140-2, consulte *Federal Information Processing Standards (FIPS)* na documentação on-line do produto IBM WebSphere MQ .
2. Este CipherSpec não pode ser usado para proteger uma conexão do WebSphere MQ Explorer com um gerenciador de filas, a menos que os arquivos de políticas irrestritos apropriados sejam aplicados ao JRE usado pelo Explorer
3. Este CipherSpec era certificado FIPS 140-2 antes de 19 de Maio de 2007.
4. Quando o WebSphere MQ é configurado para a operação compatível com FIPS 140-2, este CipherSpec pode ser usado para transferir até 32 GB de dados antes que a conexão seja finalizada com o erro AMQ9288 Para evitar esse erro, evite usar DES triplo (que foi descontinuado) ou ative a reconfiguração de chave secreta ao usar esse CipherSpec em uma configuração FIPS 140-2.

**Informações relacionadas**

[Assegurando](#)

[Integridade de dados de mensagens](#)

[Especificando CipherSpecs](#)

***XMSC\_WMQ\_SSL\_CIPHER\_SUITE***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome do CipherSuite a ser usado em uma conexão TLS para um gerenciador de filas. O protocolo usado para negociar a conexão segura depende do CipherSuite especificado.

Essa propriedade possui os seguintes valores canônicos:

- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_DES\_CBC\_SHA
- 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

Esse valor pode ser fornecido como uma alternativa para [XMSC\\_WMQ\\_SSL\\_CIPHER\\_SPEC](#).

Se um valor não vazio for especificado para [XMSC\\_WMQ\\_SSL\\_CIPHER\\_SPEC](#), esse valor substituirá a configuração para [XMSC\\_WMQ\\_SSL\\_CIPHER\\_SUITE](#). Se [XMSC\\_WMQ\\_SSL\\_CIPHER\\_SPEC](#) não tiver um valor, o valor de [XMSC\\_WMQ\\_SSL\\_CIPHER\\_SUITE](#) será usado como o conjunto de cifras a ser fornecido ao GSKit. Nesse caso, o valor é mapeado para o valor CipherSpec equivalente, conforme descrito em

“Mapeamentos de nomes CipherSuite e CipherSpec para conexões com um gerenciador de filas do IBM MQ” na página 68.

Se XMSC\_WMQ\_SSL\_CIPHER\_SPEC e XMSC\_WMQ\_SSL\_CIPHER\_SUITE estiverem vazios, o campo pChDef->SSLCipherSpec será preenchido com espaços.

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

Por padrão, a propriedade não é configurada

#### **Informações relacionadas**

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

### ***XMSC\_WMQ\_SSL\_CRYPTO\_HW***

#### **Tipo de dado:**

Sequência

#### **Propriedade de:**

ConnectionFactory

Detalhes de configuração para o hardware criptográfico conectado ao sistema do cliente.

Essa propriedade possui os seguintes valores canônicos:

- GSK\_ACCELERATOR\_RAINBOW\_CS\_OFF
- GSK\_ACCELERATOR\_RAINBOW\_CS\_ON
- GSK\_ACCELERATOR\_NCIPHER\_NF\_OFF
- GSK\_ACCELERATOR\_NCIPHER\_NF\_ON

Há um formato especial para o hardware criptográfico PKCS11 (em que DriverPath, TokenLabel e TokenPassword são sequências especificadas pelo usuário):

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

O XMS não interpreta ou altera o conteúdo da sequência. Ele copia o valor fornecido, até um limite de 256 caracteres de byte único, no MQSCO MQSCO.CryptoHardware de CryptoHardware

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

Por padrão, a propriedade não é configurada

#### **Informações relacionadas**

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

### ***XMSC\_WMQ\_SSL\_FIPS\_REQUIRED***

#### **Tipo de dado:**

Booleana

#### **Propriedade de:**

ConnectionFactory

O valor dessa propriedade determina se um aplicativo pode ou não usar conjuntos de cifras compatíveis não FIPS. Se essa propriedade for configurada como true, apenas algoritmos do FIPS serão usados para a conexão cliente-servidor.

Essa propriedade pode ter os valores a seguir, que são convertidos para os dois valores canônicos para o MQSCO MQSCO.FipsRequired:

Tabela 34. Tabela de valores para MQSCO.FlipsRequired

Value	Descrição	Valor correspondente de MQSCO.FlipsRequired
false	Qualquer CipherSpec pode ser usado	MQSSL_FIPS_NO (o padrão),
true	Somente algoritmos criptográficos certificados pelo FIPS podem ser usados no CipherSpec que se aplica a essa conexão do cliente	MQSSL_FIPS_YES

XMS copia o valor relevante no MQSCO.FlipsRequired antes de chamar MQCONNX.

O parâmetro MQSCO.FlipsRequired está disponível apenas a partir da versão 6 do WebSphere MQ. Se WebSphere MQ versão 5.3, se essa propriedade for configurada, o XMS não tentará fazer a conexão com o gerenciador de fila e, em vez disso, emitirá uma exceção apropriada.

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

#### Informações relacionadas

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

### ***XMSC\_WMQ\_SSL\_KEY\_REPOSITORY***

#### Tipo de dado:

Sequência

#### Propriedade de:

ConnectionFactory

O local do arquivo do banco de dados de chaves no qual chaves e certificados são armazenados.

XMS copia a sequência, até um limite de 256 caracteres de byte único, no MQSCO.KeyRepository do KeyRepository WebSphere MQ interpreta essa sequência como um nome de arquivo, incluindo o caminho completo..

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

Por padrão, a propriedade não é configurada.

#### Informações relacionadas

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

### ***XMSC\_WMQ\_SSL\_KEY\_RESETCOUNT***

#### Tipo de dado:

System.Int32

#### Propriedade de:

ConnectionFactory

O KeyResetCount representa o número total de bytes não criptografados enviados e recebidos dentro de uma conversa SSL antes de a chave secreta ser renegociada. O número de bytes inclui informações de controle enviadas pelo MCA.

XMS copia o valor fornecido para essa propriedade no MQSCO.KeyResetCount antes de chamar MQCONNX.

O parâmetro MQSCO.KeyRestCount está disponível apenas a partir do WebSphere MQ versão 6 Se WebSphere MQ versão 5.3, se essa propriedade for configurada, o XMS não tentará fazer a conexão com o gerenciador de fila e, em vez disso, emitirá uma exceção apropriada

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

O valor padrão dessa propriedade é zero, o que significa que chaves secretas nunca são renegociadas.

### **Informações relacionadas**

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

## ***XMSC\_WMQ\_SSL\_PEER\_NAME***

### **Tipo de dado:**

Sequência

### **Propriedade de:**

ConnectionFactory

O nome do peer a ser usado em uma conexão SSL com um gerenciador de filas.

Não há nenhuma lista de valores canônicos para esta propriedade Em vez disso, deve-se construir essa cadeia de acordo com as regras para SSLPEER

Um exemplo de nome de peer é:

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS copia a sequência na página de códigos de byte único correta e coloca os valores corretos em MQCD.SSLPeerNamePtr e MQCD.SSLPeerNameLength antes de chamar MQCONN.

Essa propriedade será relevante apenas se o aplicativo se conectar a um gerenciador de filas em modo cliente

Para .NET apenas: de IBM MQ 8.0, as conexões gerenciadas para IBM MQ (WMQ\_CM\_CLIENT) e conexões não gerenciadas para IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) suportam conexões TLS/SSL.

Por padrão, a propriedade não é configurada

### **Informações relacionadas**

[Suporte ao SSL e TLS para o cliente .NET não gerenciado](#)

[Suporte para SSL e TLS para o cliente .NET gerenciado](#)

[SSLPEERNAME](#)

## ***XMSC\_WMQ\_SYNCPOINT\_ALL\_GETS***

### **Tipo de dado:**

System.Boolean

### **Propriedade de:**

ConnectionFactory

Se todas as mensagens devem ser recuperadas de filas dentro do controle de ponto de sincronização.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
false	Quando as circunstâncias forem apropriadas, o cliente XMS poderá recuperar mensagens de filas fora do controle do ponto de sincronização..
true	O cliente XMS deve recuperar todas as mensagens das filas dentro do controle de ponto de sincronização.

O valor padrão é falso.

## ***XMSC\_WMQ\_TARGET\_CLIENT***

**Tipo de dado:**

System.Int32

**Propriedade de:**

Destino

**Nome usado em um URI:**

targetClient

Se as mensagens enviadas para o destino contêm um cabeçalho MQRFH2.

Se um aplicativo enviar uma mensagem contendo um cabeçalho MQRFH2 , o aplicativo de recepção deverá ser capaz de manipular o cabeçalho.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
XMSC_WMQ_TARGET_DEST_JMS	As mensagens enviadas ao destino contêm um cabeçalho MQRFH2 . Especifique esse valor se o aplicativo estiver enviando as mensagens para outro aplicativo XMS , um aplicativo WebSphere JMS ou um aplicativo WebSphere MQ nativo projetado para manipular um cabeçalho MQRFH2 .
XMSC_WMQ_TARGET_DEST_MQ	As mensagens enviadas ao destino não contêm um cabeçalho MQRFH2 . Especifique esse valor se o aplicativo estiver enviando as mensagens para um aplicativo WebSphere MQ nativo que não foi projetado para manipular um cabeçalho MQRFH2 .

O valor padrão é XMSC\_WMQ\_TARGET\_DEST\_JMS.

## ***XMSC\_WMQ\_TEMP\_Q\_PREFIX***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O prefixo usado para formar o nome da WebSphere MQ fila dinâmica que é criada quando o aplicativo cria um XMS fila temporária.

As regras para formar o prefixo são as mesmas que as regras para formar o conteúdo do campo **DynamicQName** em um descritor de objeto, mas o último caractere não em branco deve ser um asterisco (\*). Se a propriedade não for configurada, o valor usado será CSQ.\* on z/OS e AMQ.\* nas outras plataformas.. Por padrão, a propriedade não é configurada

Essa propriedade é relevante apenas para o domínio ponto-a-ponto

## ***XMSC\_WMQ\_TEMP\_TOPIC\_PREFIX***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory, Destino

Ao criar tópicos temporários, o XMS gera uma sequência de tópicos no formato "TEMP/TEMPTOPICPREFIX/unique\_id" ou se essa propriedade contiver o valor padrão, essa sequência, "TEMP/unique\_id", será gerada. Especificar um valor não vazio permite que as filas modelo específicas sejam definidas para criar as filas gerenciadas para assinantes de tópicos temporários criados sob essa conexão.

Qualquer sequência não nula que consiste apenas em caracteres válidos para uma sequência de tópicos IBM WebSphere MQ é um valor válido para essa propriedade.

Por padrão, essa propriedade é configurada como "" (sequência vazia).

**Nota:** Essa propriedade é relevante apenas no domínio de publicação / assinatura

### ***XMSC\_WMQ\_TEMPORARY\_MODEL***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome da fila modelo do WebSphere MQ a partir da qual uma fila dinâmica é criada quando o aplicativo cria um XMS fila temporária.

O valor padrão da propriedade é SYSTEM.DEFAULT.MODEL.QUEUE.

Essa propriedade é relevante apenas para o domínio ponto-a-ponto

### ***XMSC\_WMQ\_WILDCARD\_FORMAT***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory, Destino

Essa propriedade determina qual versão de sintaxe curinga deve ser usada.

Ao usar a publicação / assinatura com IBM WebSphere MQ '\*' e '?' são tratados como curingas Considerando que '#' e '+' são tratados como curingas ao usar publicar assinatura com IBM Integration Bus. Essa propriedade substitui a propriedade XMSC\_WMQ\_BROKER\_VERSION..

Os valores válidos para essa propriedade são:

#### **XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY**

Reconhece os curingas de nível de tópico apenas, ou seja, '#' e '+' são tratados como curingas Esse valor é igual a XMSC\_WMQ\_BROKER\_V2.

#### **XMSC\_WMQ\_WILDCARD\_CHAR\_ONLY**

Reconhece os caracteres curingas apenas, ou seja, '\*' e '?' são tratados como curingas Esse valor é igual a XMSC\_WMQ\_BROKER\_V1.

Por padrão, essa propriedade é configurada para XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY

### ***XMSC\_WPM\_BUS\_NAME***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory e Destino

**Nome usado em um URI:**

busName

Para um connection factory, o nome do barramento de integração de serviços ao qual o aplicativo se conecta ou, para um destino, o nome do barramento de integração de serviços no qual o destino existe.

Para um destino que é um tópico, essa propriedade é o nome do barramento de integração de serviços no qual o espaço de tópicos associado existe... Este espaço de tópico é especificado pelo XMSC\_WPM\_TOPIC\_SPACE de propriedade

Se a propriedade não for configurada para um destino, a fila ou o espaço de tópico associado será considerado existente no barramento de integração de serviços ao qual o aplicativo se conecta.

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_CONNECTION\_PROTOCOL***

**Tipo de dado:**  
System.Int32

**Propriedade de:**  
Conexão

O protocolo de comunicações usado para a conexão com o mecanismo do sistema de mensagens. Essa propriedade é somente leitura.

Os valores possíveis da propriedade são os seguintes:

<b>Value</b>	<b>Significado</b>
XMSC_WPM_CP_HTTP	A conexão usa HTTP sobre TCP/IP..
XMSC_WPM_CP_TCP	A conexão usa TCP/IP.

### ***XMSC\_WPM\_CONNECTION\_PROXIMIDADE***

**Tipo de dado:**  
System.Int32

**Propriedade de:**  
ConnectionFactory

A configuração de proximidade de conexão para a conexão. Essa propriedade determina quão próximo o mecanismo do sistema de mensagens ao qual o aplicativo se conecta deve estar do servidor de autoinicialização.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Configuração de proximidade de conexão..</b>
XMSC_WPM_CONNECTION_PROXIMITY_BUS	Barramento
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	Cluster
XMSC_WPM_CONNECTION_PROXIMITY_HOST	Host
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	Servidor

O valor padrão é XMSC\_WPM\_CONNECTION\_PROXIMITY\_BUS.

### ***XMSC\_WPM\_DUR\_SUB\_HOME***

**Tipo de dado:**  
Sequência

**Propriedade de:**  
ConnectionFactory

**Nome usado em um URI:**  
durableSubscriptionInício

O nome do mecanismo do sistema de mensagens no qual todas as assinaturas duráveis para uma conexão ou um destino são gerenciadas. Mensagens a serem entregues aos assinantes duráveis são armazenadas no ponto de publicação do mesmo mecanismo do sistema de mensagens.

Um lar de assinaturas duráveis deve ser especificado para uma conexão antes que um aplicativo possa criar um assinante durável que use a conexão. Qualquer valor especificado para um destino substitui o valor especificado para a conexão.

Por padrão, a propriedade não é configurada

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

### ***XMSC\_WPM\_HOST\_NAME***

**Tipo de dado:**

Sequência

**Propriedade de:**

Conexão

O nome do host ou o endereço IP do sistema que contém o mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.

### ***XMSC\_WPM\_LOCAL\_ADDRESS***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Para uma conexão com um barramento de integração de serviços, essa propriedade especifica a interface de rede local a ser usada, a porta local ou o intervalo de portas locais a serem usados ou ambos.

O valor da propriedade é uma sequência com o seguinte formato:

*[host\_name] [(low\_port) [,high\_port]]*

Os significados das variáveis são os seguintes:

***host\_name***

O nome do host ou o endereço IP da interface de rede local a ser usado para a conexão

Fornecer essas informações é necessário apenas se o sistema no qual o aplicativo está em execução tiver duas ou mais interfaces de rede e você precisar ser capaz de especificar qual interface deve ser usada para a conexão.. Se o sistema tiver apenas uma interface de rede, somente essa interface poderá ser usada Se o sistema tiver duas ou mais interfaces de rede e você não especificar qual interface deve ser usada, a interface será selecionada aleatoriamente

***low\_port***

O número da porta local a ser usada para a conexão

Se *high\_port* também for especificado, *low\_port* será interpretado como o número da porta mais baixo em um intervalo de números de portas

***high\_port***

O número da porta mais alto em um intervalo de números de porta Uma das portas no intervalo especificado deve ser usada para a conexão..

Aqui estão alguns exemplos de valores válidos da propriedade:

JÚPITER

9.20.4.98

JUPITER (1000)

9.20.4.98(1000,2000)

(1000)

(1000,2000)

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_ME\_NAME***

**Tipo de dado:**

Sequência

**Propriedade de:**

Conexão

O nome do mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.

***XMSC\_WPM\_NON\_PERSISTENT\_MAP*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O nível de confiabilidade de mensagens não persistentes que são enviadas usando a conexão.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_SEM esforço  
PERSISTENTXMSC\_WPM\_MAPPING\_EXPRESS\_NON\_  
PERSISTENTXMSC\_WPM\_MAPPING\_RELIABLE\_NON\_  
PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

**Nível de confiabilidade**

Determinado pelo nível de confiabilidade padrão especificado para a fila ou espaço de tópicos no barramento de integração de serviços

Best effort nonpersistent

Express nonpersistent

Confiável Não Persistente

Persistente confiável

Garantido Persistente

O valor padrão é XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT.

***XMSC\_WPM\_PERSISTENT\_MAP*****Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O nível de confiabilidade de mensagens persistentes que são enviadas usando a conexão.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_SEM esforço  
PERSISTENT**Nível de confiabilidade**

Determinado pelo nível de confiabilidade padrão especificado para a fila ou espaço de tópicos no barramento de integração de serviços

Best effort nonpersistent

**Valor válido**

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

**Nível de confiabilidade**

Express nonpersistent

Confiável Não Persistente

Persistente confiável

Garantido Persistente

O valor padrão é XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT.

***XMSC\_WPM\_PORT*****Tipo de dado:**

System.Int32

**Propriedade de:**

Conexão

O número da porta atendida pelo mecanismo do sistema de mensagens para o qual o aplicativo está conectado. Essa propriedade é somente leitura.

***XMSC\_WPM\_PROVIDER\_ENDPOINTS*****Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Uma sequência de um ou mais endereços de terminal de servidores de autoinicialização. Os endereços de terminal são separados por vírgulas.

Um servidor de autoinicialização é um servidor de aplicativos responsável por selecionar o mecanismo do sistema de mensagens ao qual o aplicativo se conecta. O endereço do terminal de um servidor de autoinicialização tem o seguinte formato:

*host\_name:port\_number:chain\_name*

Os significados dos componentes de um endereço de terminal são os seguintes:

***host\_name***

O nome do host ou endereço IP do sistema no qual o servidor de autoinicialização reside. Se nenhum nome do sistema central ou endereço IP for especificado, o padrão será localhost.

***port\_number***

O número da porta na qual o servidor de autoinicialização atende solicitações recebidas. Se nenhum número de porta for especificado, o padrão será 7276..

***chain\_name***

O nome de uma cadeia de transporte de autoinicialização usada pelo servidor de autoinicialização. Os valores válidos são os seguintes:

**Valor válido**

XMSC\_WPM\_BOOTSTRAP\_HTTP

XMSC\_WPM\_BOOTSTRAP\_HTTPS

XMSC\_WPM\_BOOTSTRAP\_SSL

XMSC\_WPM\_BOOTSTRAP\_TCP

**Nome da cadeia de transporte de autoinicialização**

Sistema de Mensagens BootstrapTunneled

BootstrapTunneledSecureMessaging

Sistema de Mensagens BootstrapSecure

Sistema de Mensagens BootstrapBasic

Se nenhum nome for especificado, o valor padrão será XMSC\_WPM\_BOOTSTRAP\_TCP.

Se nenhum endereço de terminal for especificado, o padrão será localhost:7276:BootstrapBasicMessaging.

### ***XMSC\_WPM\_SSL\_CIPHER\_SUITE***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome do CipherSuite a ser usado em uma conexão TLS com um mecanismo do sistema de mensagens do WebSphere Serviço Integration Bus O protocolo usado para negociar a conexão segura depende do CipherSuite especificado.

<i>Tabela 35. Opções CipherSuite para conexão com um mecanismo do sistema de mensagens do WebSphere Serviço Integration Bus</i>	
<b>Conjunto de cifras</b>	<b>Protocolo utilizado</b>
TLS_RSA_WITH_DES_CBC_SHA	TLSv1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_128_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_256_CBC_SHA	TLSv1

**Notes:**

1. TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA e TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA CipherSuites são suportados apenas no Windows ou Solaris .... (Isso é ditado pelo GSKit.).
2. TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA foi descontinuado. No entanto, ele ainda pode ser usado para transferir até 32 GB de dados antes de a conexão ser finalizada com erro AMQ9288. Para evitar esse erro, você precisará evitar o uso de DES triplo ou ativar a reconfiguração de chave secreta ao usar esse CipherSpec.

Não há nenhum padrão para essa propriedade Se você desejar usar SSL ou TLS, deverá especificar um valor para essa propriedade, caso contrário, seu aplicativo não poderá se conectar com êxito ao servidor.

### ***XMSC\_WPM\_SSL\_FIPS\_REQUIRED***

**Tipo de dado:**

Booleana

**Propriedade de:**

ConnectionFactory

O valor dessa propriedade determina se um aplicativo pode ou não usar conjuntos de cifras compatíveis não FIPS. Se essa propriedade for configurada como true, apenas algoritmos FIPS serão usados para a conexão cliente / servidor. Configurar o valor dessa propriedade como TRUE impede que o aplicativo use conjuntos de cifras não compatíveis com FIPS.

Por padrão, a propriedade é configurada como FALSE (ou seja, modo FIPS desativado).

### ***XMSC\_WPM\_SSL\_KEY\_REPOSITORY***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

Um caminho para o arquivo que é o arquivo do conjunto de chaves contendo as chaves públicas ou privadas a serem usadas na conexão segura.

A configuração da propriedade do arquivo de conjunto de chaves para o valor especial de `XMSC_WPM_SSL_MS_CERT_STORE` especifica o uso do banco de dados de chaves do Microsoft Windows . Usar o banco de dados de chave do Microsoft Windows , localizado em **Painel de Controle > Opções da Internet > Conteúdo > Certificados**, remove a necessidade de um banco de dados de arquivo de chave separado. O uso dessa constante no Windows x64 e outras plataformas não é permitido.

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_SSL\_KEYRING\_LABEL***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O certificado a ser usado ao autenticar-se com o servidor. Se nenhum valor for especificado, o certificado padrão será usado

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_SSL\_KEYRING\_PW***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

A senha do arquivo do conjunto de chaves.

Esta propriedade pode ser usada como uma alternativa para usar `XMSC_WPM_SSL_KEYRING_STASH_FILE` para configurar a senha para o arquivo de conjunto de chaves.

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_SSL\_KEYRING\_STASH\_FILE***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome de um arquivo binário contendo a senha do arquivo do repositório de chaves.

Essa propriedade pode ser usada como uma alternativa para usar `XMSC_WPM_SSL_KEYRING_PW` para configurar a senha para o arquivo de conjunto de chaves.

Por padrão, a propriedade não é configurada

### ***XMSC\_WPM\_TARGET\_GROUP***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome de um grupo de destinos de mecanismos do sistema de mensagens. A natureza do grupo de destino é determinada pelo `XMSC_WPM_TARGET_TYPE` de propriedade

Configure esta propriedade se desejar restringir a procura de um mecanismo do sistema de mensagens para um subgrupo dos mecanismos de sistema de mensagens no barramento de integração de serviços

Se você desejar que seu aplicativo possa se conectar a qualquer mecanismo do sistema de mensagens no barramento de integração de serviços, não configure essa propriedade.

Por padrão, a propriedade não é configurada

## ***XMSC\_WPM\_TARGET\_SIGNIFICATIVO***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O significado do grupo de destinos dos mecanismos do sistema de mensagens.

Os valores válidos da propriedade são os seguintes:

<b>Valor válido</b>	<b>Significado</b>
XMSC_WPM_TARGET_SIGNIFICANCE_PREFERIDO	Um mecanismo do sistema de mensagens no grupo de destino será selecionado, se um estiver disponível. Caso contrário, um mecanismo do sistema de mensagens fora do grupo de destino será selecionado, contanto que ele esteja no mesmo barramento de integração de serviços.
XMSC_WPM_TARGET_SIGNIFICANCE_REQUIRED	O mecanismo do sistema de mensagens selecionado deve estar no grupo de destinos. Se um mecanismo do sistema de mensagens no grupo de destino não estiver disponível, o processo de conexão falhará.

O valor padrão da propriedade é XMSC\_WPM\_TARGET\_SIGNIFICANCE\_PREFERED.

## ***XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O nome da cadeia de transporte de entrada que o aplicativo deve usar para se conectar a um mecanismo do sistema de mensagens.

O valor da propriedade pode ser o nome de qualquer cadeia de transporte de entrada disponível no servidor de aplicativos que hospeda o mecanismo do sistema de mensagens. A constante nomeada a seguir é fornecida para uma das cadeias de transporte de entrada predefinidas:

<b>Constante nomeada</b>	<b>Nome da cadeia de transporte</b>
XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC	Sistema de Mensagens InboundBasic

O valor padrão da propriedade é XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC.

## ***XMSC\_WPM\_TARGET\_TYPE***

**Tipo de dado:**

System.Int32

**Propriedade de:**

ConnectionFactory

O tipo do grupo de destinos de mecanismos do sistema de mensagens. Essa propriedade determina a natureza do grupo de destino identificado pela propriedade XMSC\_WPM\_TARGET\_GROUP.

Os valores válidos da propriedade são os seguintes:

**Valor válido**

XMSC\_WPM\_TARGET\_TYPE\_BUSMEMBER

**Significado**

O nome do grupo de destino é o nome de um membro do barramento O grupo de destino é todos os mecanismos do sistema de mensagens no membro do barramento

XMSC\_WPM\_TARGET\_TYPE\_CUSTOM

O nome do grupo de destino é o nome de um grupo definido pelo usuário de mecanismos do sistema de mensagens O grupo de destino é todos os mecanismos do sistema de mensagens registrados com o grupo definido pelo usuário.

XMSC\_WPM\_TARGET\_TYPE\_ME

O nome do grupo de destino é o nome de um mecanismo do mensagens. O grupo de destino é o mecanismo do sistema de mensagens especificado

Por padrão, a propriedade não é configurada

***XMSC\_WPM\_TEMP\_Q\_PREFIX***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O prefixo usado para formar o nome da fila temporária que é criada no barramento de integração de serviços quando o aplicativo cria um XMS fila provisória. O prefixo pode conter até 12 caracteres..

O nome de uma fila temporária começa com os caracteres "\_Q" seguidos pelo prefixo. O restante do nome consiste em caracteres gerados pelo sistema..

Por padrão, a propriedade não é configurada, o que significa que o nome de uma fila temporária não tem um prefixo

Essa propriedade é relevante apenas para o domínio ponto-a-ponto

***XMSC\_WPM\_TEMP\_TOPIC\_PREFIX***

**Tipo de dado:**

Sequência

**Propriedade de:**

ConnectionFactory

O prefixo usado para formar o nome de um tópico temporário que é criado pelo aplicativo. O prefixo pode conter até 12 caracteres..

O nome de um tópico temporário começa com os caracteres "\_T" seguidos pelo prefixo. O restante do nome consiste em caracteres gerados pelo sistema..

Por padrão, a propriedade não é configurada, o que significa que o nome de um tópico temporário não possui um prefixo..

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura

***XMSC\_WPM\_TOPIC\_SPACE***

**Tipo de dado:**

Sequência

**Propriedade de:**

Destino

**Nome usado em um URI:**

topicSpace

O nome do espaço de tópico que contém o tópico. Apenas um destino que é um tópico pode ter essa propriedade

Por padrão, a propriedade não é configurada, o que significa que o espaço de tópico padrão é assumido

Essa propriedade é relevante apenas para o domínio Publicação/Assinatura



## Avisos

---

Estas informações foram desenvolvidas para produtos e serviços oferecidos nos Estados Unidos.

É possível que a IBM não ofereça os produtos, serviços ou recursos discutidos nesta publicação em outros países. Consulte seu representante local do IBM para obter informações sobre produtos e serviços disponíveis atualmente em sua área. Qualquer referência a um IBM produto, programa ou serviço não se destina a estado ou significa que apenas esse produto IBM, programas ou serviços possam ser utilizados. Qualquer produto, programa ou serviço funcionalmente equivalente, que não infrinja nenhum direito de propriedade intelectual da IBM poderá ser utilizado em substituição. Entretanto, a avaliação e verificação da operação de qualquer produto, programa ou serviço não IBM são de responsabilidade do Cliente.

A IBM pode ter patentes ou aplicativos de patentes pendentes relativas aos assuntos tratados nesta publicação. O fornecimento desta publicação não garante ao Cliente nenhum sobre tais patentes. É possível enviar pedidos de licença, por escrito, para:

Relações Comerciais e Industriais da IBM  
Av. Pasteur, 138-146  
Botafogo  
Rio, RJ 10504-1785  
U.S.A.

Para pedidos de licença relacionados a informações de DBCS (Conjunto de Caracteres de Byte Duplo), entre em contato com o Departamento de Propriedade Intelectual da IBM em seu país ou envie pedidos de licença, por escrito, para:

licença de propriedade intelectual  
IBM World Trade Asia Corporation Licensing  
IBM Japan, Ltd.  
Minato-ku  
Tóquio 103-8510, Japão

**O parágrafo a seguir não se aplica a nenhum país em que tais disposições não estejam de acordo com a legislação local:** A INTERNATIONAL BUSINESS MACHINES CORPORATION FORNECE ESTA PUBLICAÇÃO "NO ESTADO EM QUE SE ENCONTRA", SEM GARANTIA DE NENHUM TIPO, SEJA EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS A ELAS NÃO SE LIMITANDO, AS GARANTIAS IMPLÍCITAS DE NÃO INFRAÇÃO, COMERCIALIZAÇÃO OU ADEQUAÇÃO A UM DETERMINADO PROPÓSITO. Alguns países não permitem a exclusão de garantias expressas ou implícitas em certas transações; portanto, essa disposição pode não se aplicar ao Cliente.

Essas informações podem conter imprecisões técnicas ou erros tipográficos. Periodicamente, são feitas nas informações aqui contidas; essas alterações serão incorporadas em futuras edições desta publicação. IBM pode aperfeiçoar e/ou alterar no produto(s) e/ou programa(s) descritos nesta publicação a qualquer momento sem aviso prévio.

Referências nestas informações a websites não IBM são fornecidas apenas por conveniência e não representam de forma alguma um endosso a esses websites. Os materiais contidos nesses websites não fazem parte dos materiais desse produto IBM e a utilização desses websites é de inteira responsabilidade do Cliente.

A IBM pode utilizar ou distribuir as informações fornecidas da forma que julgar apropriada sem incorrer em qualquer obrigação para com o Cliente.

Licenciados deste programa que desejam obter informações sobre este assunto com objetivo de permitir: (i) a troca de informações entre programas criados independentemente e outros programas (incluindo este) e (ii) a utilização mútua das informações trocadas, devem entrar em contato com:

Av. Pasteur, 138-146  
Av. Pasteur, 138-146

Botafogo  
Rio de Janeiro, RJ  
U.S.A.

Tais informações podem estar disponíveis, sujeitas a termos e condições apropriadas, incluindo em alguns casos o pagamento de uma taxa.

O programa licenciado descrito nesta publicação e todo o material licenciado disponível para ele são fornecidos pela IBM sob os termos do IBM Customer Agreement, IBM Contrato de Licença do Programa Internacional ou qualquer contrato equivalente entre as partes.

Todos os dados de desempenho aqui contidos foram determinados em um ambiente controlado. Portanto, os resultados obtidos em outros ambientes operacionais podem variar significativamente. Algumas medidas podem ter sido tomadas em sistemas em nível de desenvolvimento e não há garantia de que estas medidas serão iguais em sistemas geralmente disponíveis. Além disto, algumas medidas podem ter sido estimadas através de extrapolação. Os resultados reais podem variar. usuários deste documento devem verificar os dados aplicáveis para seu ambiente específico.

As informações relativas a produtos não IBM foram obtidas junto aos fornecedores dos respectivos produtos, de seus anúncios publicados ou de outras fontes disponíveis publicamente. A IBM não testou estes produtos e não pode confirmar a precisão de seu desempenho, compatibilidade nem qualquer outra reivindicação relacionada a produtos não IBM. Dúvidas sobre os recursos de produtos não IBM devem ser encaminhadas diretamente a seus fornecedores.

Todas as declarações relacionadas aos objetivos e intenções futuras da IBM estão sujeitas a alterações ou cancelamento sem aviso prévio e representam somente metas e objetivos.

Essas informações contêm exemplos de dados e relatórios utilizados em operações diárias de negócios. Para ilustrá-los da forma mais completa possível, os exemplos incluem nomes de indivíduos, empresas, marcas e produtos. Todos estes nomes são fictícios e qualquer semelhança com os nomes e endereços utilizados por uma empresa real é mera coincidência.

#### LICENÇA DE COPYRIGHT :

Estas informações contêm programas de aplicativos de amostra na linguagem fonte, ilustrando as técnicas de programação em diversas plataformas operacionais. O Cliente pode copiar, modificar e distribuir estes programas de amostra sem a necessidade de pagar à IBM, com objetivos de desenvolvimento, uso, marketing ou distribuição de programas aplicativos em conformidade com a interface de programação de aplicativo para a plataforma operacional para a qual os programas de amostra são criados. Esses exemplos não foram testados completamente em todas as condições. Portanto, a IBM não pode garantir ou implicar a confiabilidade, manutenção ou função destes programas.

Se estiver visualizando estas informações em formato eletrônico, as fotografias e ilustrações coloridas poderão não aparecer.

## Informações sobre a Interface de Programação

---

As informações da interface de programação, se fornecidas, destinam-se a ajudá-lo a criar software aplicativo para uso com este programa.

Este manual contém informações sobre interfaces de programação desejadas que permitem que o cliente grave programas para obter os serviços do WebSphere MQ.

No entanto, estas informações também podem conter informações sobre diagnósticos, modificações e ajustes. As informações sobre diagnósticos, modificações e ajustes são fornecidas para ajudá-lo a depurar seu software aplicativo.

**Importante:** Não use essas informações de diagnóstico, modificação e ajuste como uma interface de programação, pois elas estão sujeitas a mudanças

## Marcas comerciais

---

IBM, o logotipo IBM , ibm.com, são marcas registradas da IBM Corporation, registradas em várias jurisdições no mundo todo Uma lista atual de marcas registradas da IBM está disponível na Web em "Informações de copyright e marca registrada" [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml). Outros nomes de produtos e serviços podem ser marcas comerciais da IBM ou de outras empresas.

Microsoft e Windows são marcas comerciais da Microsoft Corporation nos Estados Unidos e/ou em outros países.

UNIX é uma marca registrada do The Open Group nos Estados Unidos e em outros países.

Linux® é uma marca registrada de Linus Torvalds nos Estados Unidos e/ou em outros países.

Este produto inclui software desenvolvido pelo Projeto Eclipse (<http://www.eclipse.org/>).

Java e todas as marcas comerciais e logotipos baseados em Java são marcas comerciais ou marcas registradas da Oracle e/ou de suas afiliadas.







Part Number:

(1P) P/N: