

7.5

*IBM Message Service Client for .NET*

**IBM**

## 注記

本書および本書で紹介する製品をご使用になる前に、[253 ページの『特記事項』](#)に記載されている情報をお読みください。

本書は、IBM® WebSphere® MQ バージョン 7 リリース 5、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様が IBM に情報を送信する場合、お客様は IBM に対し、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で情報を使用または配布する非独占的な権利を付与します。

© Copyright International Business Machines Corporation 2007 年, 2024.

# 目次

<b>Message Service Client for .NET</b> .....	<b>5</b>
Message Service Client for .NET の概要.....	5
このリリースの新機能.....	6
メッセージングのスタイル.....	7
XMS オブジェクト・モデル.....	7
オブジェクトの属性とプロパティ.....	9
管理対象オブジェクト.....	9
XMS メッセージ・モデル.....	10
アプリケーションで最新バージョン以外の XMS を使用する場合.....	11
メッセージング・サーバー環境のセットアップ.....	11
IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネー ジャーおよびブローカーの構成.....	11
ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成.....	13
WebSphere サービス統合バスに接続するアプリケーションのための サービス統合バス の構成... ..	14
インストール・ウィザードを使用した Message Service Client for .NET のインストール.....	16
IBM WebSphere MQ に接続する XMS アプリケーションの前提条件.....	18
XMS サンプル・アプリケーションの使用.....	18
サンプル・アプリケーション.....	19
サンプル・アプリケーションの実行.....	20
ドット・ネット サンプル・アプリケーションの作成.....	21
XMS アプリケーションの作成.....	22
XMS アプリケーションの書き込み.....	22
XMS .NET アプリケーションの作成.....	46
管理対象オブジェクトでの作業.....	52
XMS アプリケーションの通信の保護.....	66
XMS メッセージ.....	71
トラブルシューティング.....	85
ドット・ネット アプリケーションのトレース構成.....	85
ドット・ネット アプリケーションの FFDC 構成.....	90
トラブルシューティングのヒント.....	90
Message Service Client for .NET のリファレンス.....	91
ドット・ネットインターフェース.....	91
XMS オブジェクトのプロパティ.....	181
<b>特記事項</b> .....	<b>253</b>
プログラミング・インターフェース情報.....	254
商標.....	254



## Message Service Client for .NET の概要

---

Message Service Client for .NET は、XMS と呼ばれるアプリケーション・プログラミング・インターフェース (API) を提供します。この API は、Java Message Service (JMS) API と同じ一連のインターフェースを備えています。Message Service Client for .NET には、.NET 準拠言語で使用できる、完全に管理された XMS 実装があります。

XMS では以下がサポートされます。

- Point-to-Point ・メッセージング
- publish/subscribe ・メッセージング
- 同期メッセージ配信
- 非同期メッセージ配信

XMS アプリケーションは、以下のタイプのアプリケーションとメッセージを交換できます。

- XMS アプリケーション
- IBM WebSphere MQ classes for JMS アプリケーション
- ネイティブ IBM WebSphere MQ アプリケーション
- WebSphere デフォルト・メッセージング・プロバイダーを使用する JMS アプリケーション

XMS アプリケーションは、以下のメッセージング・サーバーに接続し、これらのサーバーのリソースを使用できます。

### IBM WebSphere MQ キュー・マネージャー

アプリケーションはバインディング・モードまたはクライアント・モードのいずれかで接続できます。

### WebSphere ApplicationServer サービス統合バス

アプリケーションは直接 TCP/IP 接続または HTTP over TCP/IP を使用できます。

### WebSphere Event Broker または WebSphere Message Broker

アプリケーションとブローカーの間では、WebSphere MQ Real-Time Transport を使用してメッセージを移送します。WebSphere MQ Multicast Transport を使用してメッセージをアプリケーションに送信することも可能です。

IBM WebSphere MQ キュー・マネージャーへ接続することで、XMS アプリケーションは IBM WebSphere MQ エンタープライズ・トランスポートを使用して WebSphere Event Broker または WebSphere Message Broker のブローカーと通信できます。あるいは、XMS アプリケーションは、IBM WebSphere MQ に接続することによってパブリッシュ/サブスクライブを実行することも可能です。

### 関連概念

[7 ページの『メッセージングのスタイル』](#)

[7 ページの『XMS オブジェクト・モデル』](#)

XMS API はオブジェクト指向インターフェースです。XMS オブジェクト・モデルは、JMS 1.1 オブジェクト・モデルが基本になっています。

[10 ページの『XMS メッセージ・モデル』](#)

XMS メッセージ・モデルは、IBM WebSphere MQ classes for JMS メッセージ・モデルと同じものです。

## Message Service Client for .NET の概要

---

Message Service Client for .NET は、XMS と呼ばれるアプリケーション・プログラミング・インターフェース (API) を提供します。この API は、Java Message Service (JMS) API と同じ一連のインターフェースを備えています。Message Service Client for .NET には、.NET 準拠言語で使用できる、完全に管理された XMS 実装があります。

XMS では以下がサポートされます。

- Point-to-Point ・メッセージング

- publish/subscribe・メッセージング
- 同期メッセージ配信
- 非同期メッセージ配信

XMS アプリケーションは、以下のタイプのアプリケーションとメッセージを交換できます。

- XMS アプリケーション
- IBM WebSphere MQ classes for JMS アプリケーション
- ネイティブ IBM WebSphere MQ アプリケーション
- WebSphere デフォルト・メッセージング・プロバイダーを使用する JMS アプリケーション

XMS アプリケーションは、以下のメッセージング・サーバーに接続し、これらのサーバーのリソースを使用できます。

#### **IBM WebSphere MQ キュー・マネージャー**

アプリケーションはバインディング・モードまたはクライアント・モードのいずれかで接続できます。

#### **WebSphere ApplicationServer サービス統合バス**

アプリケーションは直接 TCP/IP 接続または HTTP over TCP/IP を使用できます。

#### **WebSphere Event Broker または WebSphere Message Broker**

アプリケーションとブローカーの間では、WebSphere MQ Real-Time Transport を使用してメッセージを移送します。WebSphere MQ Multicast Transport を使用してメッセージをアプリケーションに送信することも可能です。

IBM WebSphere MQ キュー・マネージャーへ接続することで、XMS アプリケーションは IBM WebSphere MQ エンタープライズ・トランスポートを使用して WebSphere Event Broker または WebSphere Message Broker のブローカーと通信できます。あるいは、XMS アプリケーションは、IBM WebSphere MQ に接続することによってパブリッシュ/サブスクライブを実行することも可能です。

#### **関連概念**

[7 ページの『メッセージングのスタイル』](#)

[7 ページの『XMS オブジェクト・モデル』](#)

XMS API はオブジェクト指向インターフェースです。XMS オブジェクト・モデルは、JMS 1.1 オブジェクト・モデルが基本になっています。

[10 ページの『XMS メッセージ・モデル』](#)

XMS メッセージ・モデルは、IBM WebSphere MQ classes for JMS メッセージ・モデルと同じものです。

## **このリリースの新機能**

このリリースの Message Service Client for .NET ではいくつかの機能が強化されています。

#### **85 ページの『Message Service Client for .NET アプリケーションからのメッセージ記述子の読み取りと書き込み』**

IBM WebSphere MQ メッセージの StructId と Version 以外のすべてのメッセージ記述子フィールドにアクセスできます。BackoutCount フィールドは読み取り可能ですが、書き込むことはできません。これらのフィールドにアクセスできるのは、IBM WebSphere MQ バージョン 6 以上のキュー・マネージャーに接続する場合に限られます。アクセス制御は、後述する宛先プロパティによって行います。

#### **19 ページの『サンプル・アプリケーション』**

これらの XMS サンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを検証したりするときに活用できます。

#### **パフォーマンスの向上**

XMS .NET のパフォーマンスが改善されました。

#### **46 ページの『XMS による自動 IBM WebSphere MQ クライアント再接続』**

ネットワーク、キュー・マネージャー、サーバーで障害が発生した後に、自動的に再接続するように WebSphere MQ V7.1 XMS IBM WebSphere MQ クライアントを構成できます。

## 41 ページの『XMS による管理 IBM WebSphere MQ XA トランザクション』

XMS で管理 WebSphere MQ XA トランザクションを使用できます。

### **GMO\_CONVERT (GMO\_CONVERT)**

メッセージで GMO\_CONVERT の値を指定するかどうかは任意です。GMO\_CONVERT の値を指定する場合は、その値に基づいて変換が実行されます。

## メッセージングのスタイル

XMS は、メッセージングのスタイルとして Point-to-Point および publish/subscribe をサポートします。

メッセージングのスタイルは、メッセージング・ドメインとも呼ばれます。

### **Point-to-Point ・メッセージング**

一般的な形式の Point-to-Point メッセージングでは、キューイングを使用します。最も単純な事例では、暗黙的または明示的に宛先キューを指定することにより、アプリケーションが他のアプリケーションにメッセージを送信します。下位層のメッセージング・システムおよびキューイング・システムは、送信側アプリケーションからメッセージを受信して、そのメッセージをシステムの宛先キューに転送します。受信側アプリケーションは、受信後、キューからメッセージを取り出すことができます。

下位層のメッセージング・システムとキューイング・システムに WebSphere Message Broker が含まれている場合、WebSphere Message Broker は、メッセージを複製してメッセージのコピーを別々のキューに転送できます。その結果、複数のアプリケーションがメッセージを受信できるようになります。WebSphere Message Broker は、メッセージを変換してデータを追加することもできます。

Point-to-Point メッセージングの重要な特性は、アプリケーションがメッセージの送信時にメッセージをローカル・キューに配置することです。メッセージの送信先になる宛先キューは、下位層のメッセージング・システムとキューイング・システムによって決まります。受信側のアプリケーションは、その宛先キューからメッセージを取り出します。

### **パブリッシュ/サブスクライブ ・メッセージング**

publish/subscribe ・メッセージングでは、パブリッシャーとサブスクライバーという 2 種類のアプリケーションがあります。

パブリッシャー は、パブリケーション・メッセージの形式で情報を提供します。パブリッシャーは、メッセージを公開するときにトピックを指定します。トピックは、メッセージ内部の情報の主題を識別します。

サブスクライバー は、公開されている情報のコンシューマーです。サブスクライバーは、サブスクリプションを作成することによって、関心のあるトピックを指定します。

パブリッシュ/サブスクライブ ・システムは、パブリッシャーからパブリケーションを受け取り、サブスクライバーからサブスクリプションを受け取ります。その後、パブリケーションをサブスクライバーに送付します。サブスクライバーは、サブスクライブしたトピックについてのみパブリケーションを受け取ります。

publish/subscribe ・メッセージングの重要な特性は、パブリッシャーがメッセージのパブリッシュ時にトピックを指定することです。サブスクライバーを指定するわけではありません。サブスクライバーのないトピックにメッセージを公開した場合、このメッセージを受信するアプリケーションは存在しません。

1つのアプリケーションがパブリッシャーとサブスクライバーの両方を兼ねることもあります。

## XMS オブジェクト ・モデル

XMS API はオブジェクト指向インターフェースです。XMS オブジェクト ・モデルは、JMS 1.1 オブジェクト ・モデルが基本になっています。

次のリストでは、主な XMS クラス、つまりオブジェクトのタイプを要約しています。

### **ConnectionFactory**

ConnectionFactory オブジェクトは、接続に使用する一連のパラメーターをカプセル化します。アプリケーションは接続経路の作成に ConnectionFactory を使用します。アプリケーションは、実行時にパラメーターを提供して ConnectionFactory オブジェクトを作成できます。あるいは、管理対

象オブジェクトのリポジトリに接続パラメーターを格納しておくことも可能です。アプリケーションは、そのリポジトリからオブジェクトを取り出し、そのオブジェクトから `ConnectionFactory` オブジェクトを作成できます。

### Connection

`Connection` オブジェクトは、アプリケーションからメッセージング・サーバーへのアクティブな接続をカプセル化したオブジェクトです。アプリケーションは、接続を使用してセッションを作成します。

### Destination

アプリケーションは、`Destination` オブジェクトを使用してメッセージを送受信します。publish/subscribe・ドメインでは、`Destination` オブジェクトはトピックをカプセル化し、Point-to-Point ドメインでは、`Destination` オブジェクトはキューをカプセル化します。アプリケーションは、実行時にパラメーターを提供して `Destination` オブジェクトを作成できます。あるいは、管理対象オブジェクトのリポジトリに格納されているオブジェクト定義から `Destination` オブジェクトを作成することも可能です。

### Session

`Session` オブジェクトは、メッセージを送受信するための単一スレッド・コンテキストです。アプリケーションは、`Session` オブジェクトを使用して、`Message`、`MessageProducer`、`MessageConsumer` の各オブジェクトを作成します。

### Message

`Message` オブジェクトは、アプリケーションが `MessageProducer` オブジェクトを使用して送信し、`MessageConsumer` オブジェクトを使用して受信する `Message` オブジェクトをカプセル化したオブジェクトです。

### MessageProducer

`MessageProducer` オブジェクトは、アプリケーションが宛先にメッセージを送信するために使用するオブジェクトです。

### MessageConsumer

`MessageConsumer` オブジェクトは、アプリケーションが宛先に送信されたメッセージを受信するために使用するオブジェクトです。

8 ページの図 1 は、これらのオブジェクトとその関係を示します。

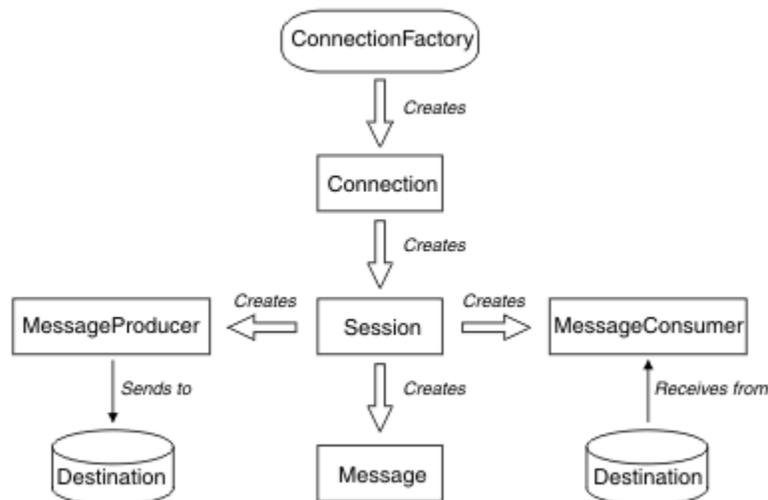


図 1. XMS オブジェクトとその関係

ドット・ネットでは、XMS クラスは一組のドット・ネット インターフェースとして定義されます。XMS ドット・ネット アプリケーションをコーディングするときに必要なのは、宣言済みのインターフェースだけです。

XMS オブジェクト・モデルは、*Java Message Service Specification*、バージョン 1.1 に記述されている、ドメインに依存しないインターフェースに基づいています。ドメイン固有のクラス (Topic、TopicPublisher、TopicSubscriber など) は提供されません。

## オブジェクトの属性とプロパティ

XMS オブジェクトには、オブジェクトの特性である属性とプロパティを設定できます。これらはさまざまな方法で実装されます。

### 属性

オブジェクトは、属性に値がない場合でも、常時存在してストレージを占有します。この点で、属性は固定長データ構造のフィールドと似ています。異なる特徴としては、属性にはそれぞれ、その値を設定および取得するための独自のメソッドがあることが挙げられます。

### プロパティ

オブジェクトのプロパティが存在してストレージを占有するのは、その値を設定した後だけです。値の設定後にプロパティを削除したり、そのストレージをリカバーしたりすることはできません。値を変更することは可能です。XMS には、プロパティ値を設定および取得するための、一連の汎用メソッドが備わっています。

### 関連概念

#### XMS プリミティブ型

XMS には、Java の 8 個のプリミティブ型 (byte、short、int、long、float、double、char、および boolean) に相当するデータ型が用意されています。これにより、XMS と JMS の間で、データの損失や破損が発生することなくメッセージを交換することができます。

#### プロパティ値のデータ型の暗黙的な変換

アプリケーションがプロパティの値を取得するときに、XMS によりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMS がその変換を実行する方法を規定します。

### 関連資料

#### アプリケーション・データの要素のデータ型

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

## 管理対象オブジェクト

管理対象オブジェクトを使用すると、クライアント・アプリケーションが使用する接続設定を中央のリポジトリから管理できます。アプリケーションは、中央のリポジトリからオブジェクト定義を取り出して使用することにより、ConnectionFactory オブジェクトや Destination オブジェクトを作成できます。管理対象オブジェクトを使用すれば、アプリケーションと、アプリケーションが実行時に使用するリソースとを切り離すことができます。

例えば、XMS アプリケーションの記述やテストは、テスト環境で一組の接続経路と宛先を参照する管理対象オブジェクトを使用して行うことができます。アプリケーションをデプロイするときには、管理対象オブジェクトを変更して、アプリケーションが実稼働環境の接続と宛先を参照するように構成できます。

XMS は、次の 2 種類の管理対象オブジェクトをサポートしています。

- ConnectionFactory オブジェクト。このオブジェクトをアプリケーションが使用する目的は、サーバーへの初期接続経路の作成です。
- Destination オブジェクト。このオブジェクトをアプリケーションが使用する目的は、送信対象メッセージの宛先と受信対象メッセージの送信元の指定です。宛先は、アプリケーションの接続先となるサーバー上のトピックまたはキューです。

管理ツール **JMSAdmin** は、IBM WebSphere MQ に付属しています。これは、管理対象オブジェクトの中央リポジトリでの管理対象オブジェクトを作成および管理するために使用されます。

リポジトリ内の管理対象オブジェクトは、IBM WebSphere MQ classes for JMS アプリケーションと XMS アプリケーションで使用できます。XMS アプリケーションは、ConnectionFactory オブジェクトと Destination オブジェクトを使用して、IBM WebSphere MQ キュー・マネージャーに接続できます。管

理者は、リポジトリ内に保持されているオブジェクト定義を、アプリケーション・コードに影響を与えずに変更できます。

次の図は、XMS アプリケーションによる管理対象オブジェクトの通常的使用方法を示しています。

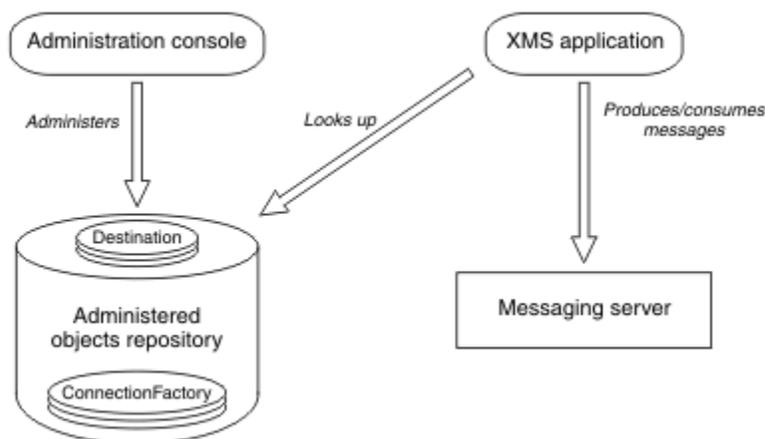


図 2. XMS アプリケーションによる管理対象オブジェクトの標準的な使用方法

### 関連概念

サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### 関連タスク

管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする

ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

## XMS メッセージ・モデル

XMS メッセージ・モデルは、IBM WebSphere MQ classes for JMS メッセージ・モデルと同じものです。

特に、XMS は、次に示すように IBM WebSphere MQ classes for JMS が実装するのと同じメッセージ・ヘッダー・フィールドおよびメッセージ・プロパティーを実装します。

- JMS ヘッダー・フィールド。これらのフィールドには、JMS というプレフィックスで始まる名前が付いています。
- JMS 定義のプロパティー。これらのフィールドには、名前が JMSX というプレフィックスで始まるプロパティーがあります。
- IBM 定義のプロパティー。これらのフィールドには、名前が JMS\_IBM\_ というプレフィックスで始まるプロパティーがあります。

このため、XMS アプリケーションは、IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できます。それぞれのメッセージには、アプリケーションによって設定されるヘッダー・フィールドとプロパティーもあれば、XMS または IBM WebSphere MQ classes for JMS によって設定されるヘッダー・フィールドとプロパティーもあります。XMS または IBM WebSphere MQ classes for JMS によって設定されるフィールドの中には、メッセージの送信時に設定されるフィールドもあれば、メッセージの受信時に設定されるフィールドもあります。適切な状況では、ヘッダー・フィールドとプロパティーがメッセージング・サーバー経由でメッセージと一緒に伝搬します。そのようにして、メッセージを受信するすべてのアプリケーションで使用できるようになります。

## アプリケーションで最新バージョン以外の XMS を使用する場合

デフォルトでは、XMS の最新バージョンがインストールされると、以前のバージョンを使用しているアプリケーションは、再コンパイルする必要なく自動的に最新バージョンに切り替えます。

### このタスクについて

複数バージョンの共存機能により、最新バージョンの XMS をインストールしても、以前のバージョン XMS が上書きされることはありません。代わりに、同じ XMS .NET アセンブリーの複数のインスタンスがグローバル・アセンブリー・キャッシュ (GAC) 内に共存しますが、そのバージョン番号は異なります。内部で、GAC は、ポリシー・ファイルを使用して、アプリケーション呼び出しを最新バージョンの XMS に転送します。アプリケーションを実行するために再コンパイルは必要ありません。最新バージョンの XMS .NET に用意されている新機能を利用することも可能です。

ただし、アプリケーションで古い XMS バージョンを使用する必要がある場合は、アプリケーション構成ファイルで `publisherpolicy` 属性を `no` に設定します。

注：アプリケーション構成ファイルは、そのファイルが関連する実行可能プログラムの名前とサフィックス `.config` から成る名前を持つファイルです。例えば、`text.exe` のアプリケーション構成ファイルの名前は `text.exe.config` になります。

ただし、いかなる時点でも、システムのアプリケーションはすべて、同じバージョンの XMS .NET を使用します。

## メッセージング・サーバー環境のセットアップ

本節章では、XMS アプリケーションがサーバーに接続できるように、メッセージング・サーバー環境をセットアップする方法を説明します。

IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションでは、IBM WebSphere MQ クライアント (バインディング・モードの場合はキュー・マネージャー) が必要です。

ブローカーとのリアルタイム接続を使用するアプリケーションには、現在のところ前提条件はありません。

XMS アプリケーション (XMS に付属のサンプル・アプリケーションを含む) を実行するには、その前にメッセージング・サーバー環境をセットアップする必要があります。

この節章は、以下のトピック節から構成されます。

- [11 ページの『IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成』](#)
- [13 ページの『ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成』](#)
- [14 ページの『WebSphere サービス統合バス に接続するアプリケーションのための サービス統合バスの構成』](#)

### 関連タスク

[インストール・ウィザードを使用した Message Service Client for .NET のインストール](#)  
インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

## IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成

このセクションでは、IBM WebSphere MQ バージョン 7.0 を使用していることを前提としています。IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションを実行するには、その前にキュー・マネージャーを構成する必要があります。publish/subscribe・アプリケーションでは、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合に追加構成が必要です。

## 始める前に

XMS は、WebSphere Message Broker バージョン 6.1 以前で動作します。

この作業を開始する前に、以下の手順を実行してください。

- アプリケーションが、稼働中のキュー・マネージャーにアクセスできることを確認します。
- アプリケーションがパブリッシュ/サブスクライブ・アプリケーションであり、キュー・パブリッシュ/サブスクライブ・インターフェースを使用する場合は、キュー・マネージャーの「PSMODE」属性が「ENABLED」に設定されていることを確認してください。
- アプリケーションが使用する接続ファクトリーでキュー・マネージャーへ接続するようにプロパティが適切に設定されていることを確認します。アプリケーションがパブリッシュ/サブスクライブ・アプリケーションである場合は、適切な接続ファクトリー・プロパティがブローカーを使用するように設定されていることを確認します。接続ファクトリーのプロパティの詳細については、[183 ページの『ConnectionFactory のプロパティ』](#)を参照してください。

## このタスクについて

WebSphere MQ JMS アプリケーションを実行するようにキュー・マネージャーおよびキュー・パブリッシュ/サブスクライブ・インターフェースを構成するのと同じ方法で、XMS アプリケーションを実行するようにキュー・マネージャー およびブローカーを構成します。以下のステップで、実行する必要のある操作を要約しています。

## 手順

1. キュー・マネージャーで、アプリケーションが必要とするキューを作成します。

キューの作成方法については、*WebSphere MQ* 製品資料の「キューの定義」トピックを参照してください。

アプリケーションが publish/subscribe・アプリケーションであり、IBM WebSphere MQ classes for JMS システム・キューへのアクセスを必要とするキュー・パブリッシュ/サブスクライブ・インターフェースを使用する場合は、ステップ 4a でキューを作成します。

2. アプリケーションに関連付けられたユーザー ID に、キュー・マネージャーに接続する権限と、キューにアクセスするための適切な権限を付与します。

許可については、*IBM WebSphere MQ* 製品資料の「セキュリティー」セクションを参照してください。アプリケーションがクライアント・モードでキュー・マネージャーに接続する場合は、*IBM WebSphere MQ* 製品資料の「クライアント」トピックも参照してください。

3. アプリケーションがクライアント・モードでキュー・マネージャーに接続する場合は、キュー・マネージャーでサーバー接続チャンネルが定義されていること、およびリスナーが開始されたことを確認してください。

これを行う方法については、*IBM WebSphere MQ* 製品資料の「クライアント」トピックを参照してください。

キュー・マネージャーに接続するアプリケーションごとにこのステップを実行する必要はありません。1つのサーバー接続チャンネル定義および1つのリスナーが、クライアント・モードで接続するすべてのアプリケーションをサポートします。

4. アプリケーションが publish/subscribe・アプリケーションで、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合は、以下のステップを実行します。

- a) キュー・マネージャーで、IBM WebSphere MQ で提供される MQSC コマンドのスクリプトを実行して、IBM WebSphere MQ classes for JMS システム・キューを作成します。WebSphere Message Broker に関連するユーザー ID がキューにアクセスする権限を持っていることを確認してください。

スクリプトの場所と実行方法については、*WebSphere MQ* 製品資料の「Java™の使用」トピックを参照してください。

キュー・マネージャーでこの手順を 1 回だけ実行します。同じ IBM WebSphere MQ classes for JMS システム・キューのセットが、キュー・マネージャーに接続するすべての XMS および IBM WebSphere MQ classes for JMS アプリケーションをサポートできます。

- b) アプリケーションに関連するユーザー ID に、IBM WebSphere MQ classes for JMS システム・キューにアクセスする権限を与えます。

ユーザー ID が必要とする権限については、*IBM WebSphere MQ* 製品資料の「Java の使用」トピックを参照してください。

- c) WebSphere Event Broker または WebSphere Message Broker のブローカーの場合は、アプリケーションが公開するメッセージを送信するキューを保守するためのメッセージ・フローを、作成およびデプロイします。

基本的なメッセージ・フローは、公開されたメッセージを読み取る MQInput メッセージ処理ノードと、メッセージを公開する Publication メッセージ処理ノードで構成されています。

メッセージ・フローを作成およびデプロイする方法については、WebSphere Event Broker または WebSphere Message Broker 製品資料を参照してください。

適切なメッセージ・フローが既にブローカーでデプロイされている場合は、このステップを実行する必要はありません。

## タスクの結果

これで、アプリケーションを開始することができます。

### 関連タスク

[ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成](#)

ブローカーへのリアルタイム接続を使用するアプリケーションを実行するには、まずそのブローカーを構成する必要があります。

[WebSphere サービス統合バスに接続するアプリケーションのための サービス統合バスの構成](#)

WebSphere サービス統合バスに接続するアプリケーションを実行するには、その前に、デフォルトのメッセージング・プロバイダーを使用する JMS アプリケーションを実行するようにサービス統合バスを構成する方法と同じ方法で、サービス統合バスを構成する必要があります。

[インストール・ウィザードを使用した Message Service Client for .NET のインストール](#)

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

### 関連資料

[IBM WebSphere MQ に接続する XMS アプリケーションの前提条件](#)

XMS アプリケーションで IBM WebSphere MQ に接続する場合には、いくつかの前提条件が適用されます。

## ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成

ブローカーへのリアルタイム接続を使用するアプリケーションを実行するには、まずそのブローカーを構成する必要があります。

### 始める前に

XMS は、WebSphere Message Broker バージョン 6.1 以前で動作します。

この作業を開始する前に、以下の手順を実行してください。

- アプリケーションが、稼働中のブローカーにアクセスできることを確認します。
- アプリケーションが使用する接続ファクトリーで、プロパティーがブローカーへのリアルタイム接続用に適切に設定されていることを確認します。接続ファクトリーのプロパティーについては、[183 ページの『ConnectionFactory のプロパティー』](#)を参照してください。

### このタスクについて

XMS アプリケーションを実行するようにブローカーを構成する場合と同様に、IBM WebSphere MQ classes for JMS アプリケーションを実行するようにブローカーを構成します。以下のステップで、実行する必要の

ある操作を要約していますが、詳細については、WebSphere Event Broker、または WebSphere Message Broker 製品資料を参照してください。

## 手順

1. ブローカーがメッセージを listen および公開する TCP/IP ポートからメッセージを読み取るためのメッセージ・フローを作成およびデプロイします。

以下のいずれかの方法で、これを実行することができます。

- **Real-timeOptimizedFlow** メッセージ処理ノードを含むメッセージ・フローを作成します。
- **Real-timeInput** メッセージ処理ノードおよび Publication メッセージ処理ノードを含むメッセージ・フローを作成します。

リアルタイム接続に使用されるポートを listen するには、**Real-timeOptimizedFlow** または **Real-timeInput** ノードを構成する必要があります。XMS では、リアルタイム接続のデフォルト・ポート番号は 1506 です。

適切なメッセージ・フローが既にブローカーでデプロイされている場合は、このステップを実行する必要はありません。

2. WebSphere MQ Multicast Transport を使用してアプリケーションにメッセージが配信されるようにする必要がある場合は、マルチキャストを使用可能にするようにブローカーを構成します。信頼性の高いマルチキャストが必要なトピックに、信頼性の高いサービス品質を指定して、マルチキャストを有効にする必要があるトピックを構成します。
3. アプリケーションがブローカーへの接続時にユーザー ID およびパスワードを提供し、ブローカーがこの情報を使用してアプリケーションを認証するようにしたい場合は、単純な telnet に類似したパスワード認証を行うようにユーザー・ネーム・サーバーおよびブローカーを構成します。

## タスクの結果

これで、アプリケーションを開始することができます。

### 関連タスク

[IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成](#)

このセクションでは、IBM WebSphere MQ バージョン 7.0 を使用していることを前提としています。IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションを実行するには、その前にキュー・マネージャーを構成する必要があります。publish/subscribe・アプリケーションでは、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合に追加構成が必要です。

[WebSphere サービス統合バスに接続するアプリケーションのためのサービス統合バスの構成](#)

WebSphere サービス統合バスに接続するアプリケーションを実行するには、その前に、デフォルトのメッセージング・プロバイダーを使用する JMS アプリケーションを実行するようにサービス統合バスを構成する方法と同じ方法で、サービス統合バスを構成する必要があります。

[インストール・ウィザードを使用した Message Service Client for .NET のインストール](#)

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

### 関連資料

[IBM WebSphere MQ に接続する XMS アプリケーションの前提条件](#)

XMS アプリケーションで IBM WebSphere MQ に接続する場合には、いくつかの前提条件が適用されます。

## WebSphere サービス統合バスに接続するアプリケーションのためのサービス統合バスの構成

WebSphere サービス統合バスに接続するアプリケーションを実行するには、その前に、デフォルトのメッセージング・プロバイダーを使用する JMS アプリケーションを実行するようにサービス統合バスを構成する方法と同じ方法で、サービス統合バスを構成する必要があります。

## 始める前に

このタスクを開始する前に、以下のステップを実行する必要があります。

- メッセージング・バスが作成されたこと、およびサーバーがバス・メンバーとしてバスに追加されたことを確認します。
- アプリケーションが、稼働中のメッセージング・エンジンを少なくとも 1 つ含むサービス統合バスにアクセスできることを確認します。
- HTTP オペレーションが必要である場合、HTTP メッセージング・エンジンのインバウンド・トランスポート・チャンネルを定義する必要があります。SSL と TCP のチャンネルは、デフォルトで、サーバーのインストール時に定義されます。
- アプリケーションが使用する接続ファクトリーで、ブートストラップ・サーバーを使用してサービス統合バスへ接続するようにプロパティが適切に設定されていることを確認します。必要最小限の情報は、以下のとおりです。
  - プロバイダー・エンドポイント。これは、メッセージング・サーバーへの (すなわちブートストラップ・サーバーを通じた) 接続をネゴシエーションする際に使用するロケーションおよびプロトコルを記述します。デフォルト設定でインストールしたサーバーに関する最も単純な形式では、プロバイダー・エンドポイントをサーバーのホスト名に設定できます。
  - メッセージの送信時に通過するバスの名前。

接続ファクトリーのプロパティについて詳しくは、[183 ページの『ConnectionFactory のプロパティ』](#)を参照してください。

## このタスクについて

キューまたはトピック・スペースが必要であれば、それを定義する必要があります。デフォルトでは、Default.Topic.Space というトピック・スペースがサーバーのインストール時に定義されますが、さらにトピック・スペースが必要な場合は、自分でそれらのトピック・スペースを作成する必要があります。トピック・スペース内の個々のトピックを事前に定義する必要はありません。サーバーが必要に応じて動的に個々のトピックをインスタンス化します。

以下のステップで、実行する必要がある操作を要約しています。

## 手順

1. Point-to-Point メッセージングのためにアプリケーションが必要とするキューを作成します。
2. publish/subscribe ・メッセージングのためにアプリケーションが必要とする追加トピック・スペースを作成します。

## タスクの結果

これで、アプリケーションを開始することができます。

### 関連タスク

[IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成](#)

このセクションでは、IBM WebSphere MQ バージョン 7.0 を使用していることを前提としています。IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションを実行するには、その前にキュー・マネージャーを構成する必要があります。publish/subscribe ・アプリケーションでは、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合に追加構成が必要です。

[ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成](#)

ブローカーへのリアルタイム接続を使用するアプリケーションを実行するには、まずそのブローカーを構成する必要があります。

[インストール・ウィザードを使用した Message Service Client for .NET のインストール](#)

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2 つのセットアップ・オプションが使用できます。

## 関連資料

IBM WebSphere MQ に接続する XMS アプリケーションの前提条件

XMS アプリケーションで IBM WebSphere MQ に接続する場合には、いくつかの前提条件が適用されます。

# インストール・ウィザードを使用した Message Service Client for .NET のインストール

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

## このタスクについて

Message Service Client for .NET を Windows にインストールするには、以下の手順を実行します。

## 手順

- SupportPac からインストールする場合、以下のステップを完了してください。それ以外の場合は、直接ステップ [16 ページの『2』](#)に進んでください。
  - Windows では、管理者としてログオンします。
  - dotNETClientsetup.exe インストーラーを実行します。
- インストール・ウィザードが開いて以下のメッセージを表示するまで待機します。

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

「次へ」をクリックします。

ウィザードにより、ご使用条件を読むよう求められる場合があります。

- ご使用条件を読むことを求められたときに、ご使用条件の条項を受け入れる場合は、「**使用条件の条項に同意します**」をクリックしてから、「次へ」をクリックします。

インストール・ウィザードから、ユーザーの要求に最適なセットアップ・タイプを選択するよう求められます。

- 以下のようにして必要なセットアップのタイプを選択します。
  - すべてのプログラム・フィーチャーをインストールし、それらをデフォルト・インストール・ディレクトリーにインストールするには、「完了」をクリックします。
  - インストールするフィーチャーを選択し、それらのインストール先を指定するには、「カスタム」をクリックします。
- 「次へ」をクリックします。

完全インストール・オプションを選択した場合、インストール・ウィザードには、ステップ [17 ページの『8』](#)に説明されているように、インストールを開始する準備が整ったことを示すメッセージが表示されます。カスタム・インストール・オプションを選択した場合、インストール・ウィザードからインストールするフィーチャーを尋ねられるので、ステップ [16 ページの『6』](#) およびステップ [17 ページの『7』](#) を完了してから、ステップ [17 ページの『8』](#) に進む必要があります。

- カスタム・インストールの場合のみ、フィーチャーのリスト内のアイコンをクリックして、Message Service Client for .NET フィーチャーのインストール方法に対する変更を指定します。提示されたディレクトリーに Message Service Client for .NET をインストールしない場合は、別のディレクトリーを選択します。

現在存在しないディレクトリーに Message Service Client for .NET をインストールする場合、インストール・ウィザードによりそのディレクトリーが作成されます。

XMS アプリケーションを開発する場合は、必ず「**開発ツールおよびサンプル (Development Tools and Samples)**」フィーチャーを選択してください。このフィーチャーには、サンプル・アプリケーションのほかに、ドット・ネット アプリケーションをコンパイルするときに必要なライブラリーや他のファイ

ルが用意されています。このフィーチャーを選択しない場合は、XMS アプリケーションを実行するために必要なファイルだけがインストールされます。

7. カスタム・インストール・オプションを使用する場合は、必要なオプションを選択してから、「次へ」をクリックします(手順 16 ページの『6』を参照)。

インストール・ウィザードに、インストールを開始する準備が整ったことを示すメッセージが表示されます。

8. 「インストール (Install)」をクリックして、インストールを開始します。

インストール・ウィザードに、インストールの進行状況を示すバーが表示されます。進行状況表示バーが完了するまで待機します。インストールが正常に完了すると、ウィンドウに次のメッセージが表示されます。

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

9. 「終了」をクリックしてインストール・ウィザードを閉じます。

## タスクの結果

Message Service Client for .NET が正常にインストールされ、使用できるようになります。

## 次のタスク

XMS アプリケーション (XMS に用意されているサンプル・アプリケーションも含む) を実行する前に、メッセージング・サーバー環境をセットアップする必要があります。詳細については、11 ページの『メッセージング・サーバー環境のセットアップ』を参照してください。

### 関連概念

#### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### メッセージング・サーバー環境のセットアップ

本節章では、XMS アプリケーションがサーバーに接続できるように、メッセージング・サーバー環境をセットアップする方法を説明します。

#### XMS サンプル・アプリケーションの使用

XMS に用意されているサンプル・アプリケーションは、インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。これらのサンプルで、各 API の一般的な機能の概要を確認できます。

### 関連タスク

#### IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成

このセクションでは、IBM WebSphere MQ バージョン 7.0 を使用していることを前提としています。IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションを実行するには、その前にキュー・マネージャーを構成する必要があります。publish/subscribe・アプリケーションでは、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合に追加構成が必要です。

#### ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成

ブローカーへのリアルタイム接続を使用するアプリケーションを実行するには、まずそのブローカーを構成する必要があります。

#### WebSphere サービス統合バスに接続するアプリケーションのためのサービス統合バスの構成

WebSphere サービス統合バスに接続するアプリケーションを実行するには、その前に、デフォルトのメッセージング・プロバイダーを使用する JMS アプリケーションを実行するようにサービス統合バスを構成する方法と同じ方法で、サービス統合バスを構成する必要があります。

### 関連資料

#### IBM WebSphere MQ に接続する XMS アプリケーションの前提条件

XMS アプリケーションで IBM WebSphere MQ に接続する場合には、いくつかの前提条件が適用されます。

## IBM WebSphere MQ に接続する XMS アプリケーションの前提条件

XMS アプリケーションで IBM WebSphere MQ に接続する場合には、いくつかの前提条件が適用されます。

IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションの場合、XMS アプリケーションを実行するために使用するマシンに、適切な IBM WebSphere MQ クライアント・ライブラリーをインストールする必要があります。それらのライブラリーは、ローカル・キュー・マネージャーと共にマシンにプリインストールされます。

.NET 用の XMS クライアントの場合は、IBM WebSphere MQ バージョン 7.0.1.0 またはそれ以降に付属のクライアント・ライブラリーを使用してください。これらはドット・ネットの *IBM WebSphere MQ* クラスです。これらにより、IBM WebSphere MQ バージョン 7.0、IBM WebSphere MQ バージョン 6.0、および IBM WebSphere MQ バージョン 5.3 キュー・マネージャーへのクライアント・モード接続、およびローカル・キュー・マネージャーへのバインディング・モード接続 (ローカル・キュー・マネージャーもバージョン 7.0.1.0 またはそれ以降の場合) が可能になります。

Microsoft .NET Framework Version 2.0 再頒布可能パッケージが、XMS のインストール先のコンピューターにインストールされていることが必要です。このパッケージを使用できないと、XMS のインストールは失敗します。その場合は、インストール手順を終了し、Microsoft .NET Framework Version 2.0 再頒布可能パッケージをコンピューターにインストールしてから、インストール手順を再び実行してください。

Microsoft のダウンロード・サイトで、Microsoft .NET Framework Version 2.0 再頒布可能パッケージ (x86) に対する dotnetfx.exe、および Microsoft .NET Framework Version 2.0 再頒布可能パッケージ (x64) に対する NetFx64.exe のどちらか該当する方のファイルを探す必要があります。

### 関連概念

#### [11 ページの『メッセージング・サーバー環境のセットアップ』](#)

本節章では、XMS アプリケーションがサーバーに接続できるように、メッセージング・サーバー環境をセットアップする方法を説明します。

### 関連タスク

#### [IBM WebSphere MQ キュー・マネージャーに接続するアプリケーション用のキュー・マネージャーおよびブローカーの構成](#)

このセクションでは、IBM WebSphere MQ バージョン 7.0 を使用していることを前提としています。IBM WebSphere MQ キュー・マネージャーに接続するアプリケーションを実行するには、その前にキュー・マネージャーを構成する必要があります。publish/subscribe・アプリケーションでは、キュー・パブリッシュ/サブスクライブ・インターフェースを使用している場合に追加構成が必要です。

#### [ブローカーへのリアルタイム接続を使用するアプリケーション用のブローカーの構成](#)

ブローカーへのリアルタイム接続を使用するアプリケーションを実行するには、まずそのブローカーを構成する必要があります。

#### [WebSphere サービス統合バスに接続するアプリケーションのための サービス統合バスの構成](#)

WebSphere サービス統合バスに接続するアプリケーションを実行するには、その前に、デフォルトのメッセージング・プロバイダーを使用する JMS アプリケーションを実行するようにサービス統合バスを構成する方法と同じ方法で、サービス統合バスを構成する必要があります。

#### [インストール・ウィザードを使用した Message Service Client for .NET のインストール](#)

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

## XMS サンプル・アプリケーションの使用

XMS に用意されているサンプル・アプリケーションは、インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。これらのサンプルで、各 API の一般的な機能の概要を確認できます。

### 関連概念

#### [19 ページの『サンプル・アプリケーション』](#)

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

## 関連タスク

インストール・ウィザードを使用した Message Service Client for .NET のインストール  
インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

### 20 ページの『サンプル・アプリケーションの実行』

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

### 21 ページの『ドット・ネット サンプル・アプリケーションの作成』

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

## サンプル・アプリケーション

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

独自のアプリケーションを作成するために手助けが必要な場合は、サンプル・アプリケーションを開始点として使用できます。各アプリケーションのソース・バージョンとコンパイル・バージョンの両方が提供されています。サンプルのソース・コードを検討し、アプリケーションに必要な各オブジェクト (ConnectionFactory、Connection、Session、Destination、さらに Producer と Consumer のいずれかまたは両方) を作成するための主な手順や、アプリケーションの動作を指定するときに必要な特定のプロパティを設定するための主な手順を見極めます。詳細については、22 ページの『XMS アプリケーションの書き込み』を参照してください。これらのサンプルは、XMS の将来のリリースで変更される可能性があります。

XMS に用意されている 3 セット (API ごとに 1 セットずつ) のサンプル・アプリケーションを以下の表にまとめます。

サンプル名	説明
SampleConsumerCS	キューからメッセージを取り込んだり、トピックにサブスクライブしたりするメッセージ・コンシューマー・アプリケーション。
SampleProducerCS	キューまたはトピックへのメッセージを作成するメッセージ・プロデューサー・アプリケーション。
SampleConfigCS	ファイル・ベースの管理対象オブジェクト・リポジトリを作成するために使用できる構成アプリケーション。このアプリケーションには、特定の接続設定の接続ファクトリーと宛先が格納されています。この管理対象オブジェクト・リポジトリは、各サンプル・コンシューマー・アプリケーションおよびサンプル・プロデューサー・アプリケーションで使用できます。

さまざまな API の同一の機能をサポートするサンプルには、構文上の違いがあります。

- サンプルのメッセージ・コンシューマー・アプリケーションとメッセージ・プロデューサー・アプリケーションはどちらも、以下の機能をサポートしています。
  - IBM WebSphere MQ、WebSphere Event Broker、WebSphere Message Broker (ブローカーへのリアルタイム接続を使用)、および WebSphere サービス統合バス への接続
  - 初期コンテキスト・インターフェースによる管理対象オブジェクト・リポジトリの検索
  - キューへの接続 (IBM WebSphere MQ および WebSphere サービス統合バス) およびトピックへの接続 (IBM WebSphere MQ、ブローカーへのリアルタイム接続、および WebSphere サービス統合バス)
  - ベース、バイト、マップ、オブジェクト、ストリーム、およびテキストの各メッセージ
- サンプル・メッセージ・コンシューマー・アプリケーションは、同期受信モード、非同期受信モード、および SQL セレクター・ステートメントをサポートしています。

- サンプル・メッセージ・プロデューサー・アプリケーションでは、永続送達モードと非永続送達モードがサポートされています。

## 動作モード

サンプルは以下のいずれかのモードで作動します。

### シンプル・モード

最小限のユーザー入力でサンプルを実行できます。

### 拡張モード

サンプルの動作を詳細にカスタマイズできます。

すべてのサンプルは相互に互換性があるため、異なる言語間で操作できます。

## サンプルのインストール先

Message Service Client for .NET 用のサンプル・アプリケーションがインストールされている場所を確認するには、オンラインの IBM WebSphere MQ 製品資料で「*Directories installed on Windows (.NET)*」を参照してください。

### 関連概念

[45 ページの『ユーザー独自のアプリケーションの作成』](#)

ユーザー独自のアプリケーションをビルドする方法は、サンプル・アプリケーションをビルドする場合と同様です。

### 関連タスク

[サンプル・アプリケーションの実行](#)

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

[ドット・ネット サンプル・アプリケーションの作成](#)

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

[20 ページの『サンプル・アプリケーションの実行』](#)

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

[21 ページの『ドット・ネット サンプル・アプリケーションの作成』](#)

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

## サンプル・アプリケーションの実行

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

### 始める前に

付属のサンプル・アプリケーションを実行する前に、最初にメッセージング・サーバー環境をセットアップし、アプリケーションがサーバーに接続できるようにする必要があります。[11 ページの『メッセージング・サーバー環境のセットアップ』](#)を参照してください。

### 手順

.NET サンプル・アプリケーションを実行するための手順は、以下のとおりです。

**ヒント:** サンプル・アプリケーションを実行している場合、次のように入力します。いつでも、次に何をすべきかについての支援を得ることができます。

1. サンプル・アプリケーションを実行するときのモードを選択します。

Advanced または Simple と入力します。

2. 質問に回答します。

質問の最後に大括弧で表示されるデフォルト値を選択するには、Enter を押します。別の値を選択するには、適切な値を入力してから Enter を押します。

質問の例を示します。

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

この場合、デフォルト値は wpm です (WebSphere サービス統合バスへの接続)。

## タスクの結果

サンプル・アプリケーションを実行すると、応答ファイルが現在の作業ディレクトリーに自動生成されます。応答ファイル名の形式は、`connection_type-sample_type.rsp` です (例: `wpm-producer.rsp`)。必要に応じて、生成した応答ファイルを使用して、同じオプションでサンプル・アプリケーションを再実行することも可能です。そのときに、オプションを再び入力する必要はありません。

### 関連概念

#### サンプル・アプリケーション

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

#### [19 ページの『サンプル・アプリケーション』](#)

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

### 関連タスク

#### ドット・ネット サンプル・アプリケーションの作成

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

#### [21 ページの『ドット・ネット サンプル・アプリケーションの作成』](#)

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

## ドット・ネット サンプル・アプリケーションの作成

サンプルのドット・ネット アプリケーションを作成する場合は、選択したサンプルの実行可能バージョンが作成されます。

### 始める前に

該当するコンパイラーをインストールします。オンラインの IBM IBM WebSphere MQ 製品資料で「*Message Service Client for .NET*」を参照してください。ここでは、Visual Studio 2005 がインストールされている環境で、Visual Studio 2005 の使用に慣れているユーザーが作業を行うことを前提にして、手順を説明します。

### 手順

ドット・ネット サンプル・アプリケーションをビルドするための手順は、以下のとおりです。

1. ドット・ネット サンプルに付属する `Samples.sln` ソリューション・ファイルをクリックします。
2. 「ソリューション エクスプローラ」ウィンドウで `Samples` というソリューションを右クリックし、「ソリューションのビルド」を選択します。

## タスクの結果

選択した構成に応じて、サンプルのサブフォルダー bin/Debug または bin/Release に実行可能プログラムが作成されます。このプログラムには、サフィックス CS を付けて、フォルダーと同じ名前が付けられます。例えば、メッセージ・プロデューサー・サンプル・アプリケーションの C# バージョンを作成する場合は、SampleProducer フォルダーに SampleProducerCS.exe が作成されます。

### 関連概念

#### サンプル・アプリケーション

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

#### 19 ページの『サンプル・アプリケーション』

これらのサンプル・アプリケーションは、各 API の一般的な機能の概要を示します。インストール環境やメッセージング・サーバーのセットアップを検証したり、ユーザー独自のアプリケーションを作成したりするときに活用できます。

#### 45 ページの『ユーザー独自のアプリケーションの作成』

ユーザー独自のアプリケーションをビルドする方法は、サンプル・アプリケーションをビルドする場合と同様です。

### 関連タスク

#### サンプル・アプリケーションの実行

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

#### 20 ページの『サンプル・アプリケーションの実行』

ドット・ネット サンプル・アプリケーションは、シンプル・モードまたは拡張モードで対話式に実行できます。自動生成の応答ファイルまたはカスタマイズした応答ファイルを使用して非対話式に実行することも可能です。

## XMS アプリケーションの作成

---

この節章では、XMS アプリケーションを作成する場合に役立つと思われる情報を提供します。

この節章の情報は、ドット・ネット アプリケーションを対象としています。

XMS アプリケーションの作成については、以下のトピックを参照してください。

## XMS アプリケーションの書き込み

この節章では、XMS アプリケーションを作成する場合に役立つ情報を提供します。

この節章では、XMS アプリケーションを作成する場合の一般的な概念について説明します。.NET アプリケーションの作成に固有の情報については、[46 ページの『XMS .NET アプリケーションの作成』](#)を参照してください。

この節章は、以下のトピック節から構成されます。

- [23 ページの『スレッド化モデル』](#)
- [24 ページの『ConnectionFactory オブジェクトと Connection オブジェクト』](#)
- [27 ページの『セッション数』](#)
- [31 ページの『宛先』](#)
- [36 ページの『メッセージ・プロデューサー』](#)
- [36 ページの『メッセージ・コンシューマー』](#)
- [40 ページの『キュー・ブラウザー』](#)
- [40 ページの『リクエスター』](#)
- [40 ページの『オブジェクトの削除』](#)
- [42 ページの『XMS プリミティブ型』](#)

- [42 ページの『プロパティ値のデータ型の暗黙的な変換』](#)
- [45 ページの『イテレーター』](#)
- [45 ページの『コード化文字セット ID』](#)
- [45 ページの『XMS エラーおよび例外コード』](#)
- [45 ページの『ユーザー独自のアプリケーションの作成』](#)

## 関連資料

[ドット・ネットインターフェース](#)

このトピック節では、ドット・ネット クラスのインターフェースとそのプロパティおよびメソッドについて説明します。

## スレッド化モデル

マルチスレッド・アプリケーションがどのように XMS オブジェクトを使用できるかは、一般的な規則によって決まります。

- 複数のスレッドで並行して使用できるオブジェクトは以下のタイプのオブジェクトのみです。
  - ConnectionFactory
  - 接続
  - ConnectionMetaData
  - Destination
- Session オブジェクトは、一度に 1 つのスレッドでのみ使用可能です。

このルールの例外は、API リファレンスの章 [91 ページの『Message Service Client for .NET のリファレンス』](#)で、メソッドのインターフェース定義で「スレッド・コンテキスト ("Thread context")」というラベルが付いている項目により示されています。

## 実行時に処理できるエラー状態

API 呼び出しからの戻りコードは、実行時に処理できるエラー状態です。このタイプのエラーを処理する方法は、C API または C++ API のどちらを使用しているかによって異なります。

## 実行時にエラーを検出する方法

アプリケーションが C API 関数を呼び出して、その呼び出しが失敗した場合、XMS\_OK 以外の戻りコードの応答と、失敗の原因に関する詳細情報が含まれている XMS エラー・ブロックが戻されます。

メソッドの使用時に、C++ API は例外をスローします。

アプリケーションは例外リスナーを使用して、接続の問題に関する通知を非同期に受信します。例外リスナーは XMS C または C++ API に提供されており、これらの API を使用して初期化されます。

## 実行時のエラー処理方法

一部のエラー状態が発生した場合、一部のリソースが使用不可になっており、アプリケーションが実行できるアクションは、アプリケーションが呼び出す XMS 関数によって異なります。例えば、サーバーに接続できない場合、アプリケーションは接続が確立するまで定期的に再試行します。XMS エラー・ブロックまたは例外には、実行するアクションを判別するために十分な情報が含まれていないことがあります。また、このような状況では、より詳細な診断情報が含まれているリンク・エラー・ブロックまたは例外にリンクしていることがあります。

C API では、常に XMS\_OK 以外の戻りコードの応答をテストし、常に API 呼び出しにエラー・ブロックを受け渡します。通常、実行されるアクションは、アプリケーションが使用する API 関数によって決まります。

C++ API では、常にメソッド呼び出しを try ブロックに組み込み、またすべてのタイプの XMS 例外をキャッチするため、catch 構成体に Exception クラスを指定します。

例外リスナーは、いつでも開始可能な非同期エラー状態パスです。例外リスナー関数がリスナー自体のスレッドで開始する場合、通常これは、標準の XMS API エラー状態よりも重大な障害が発生していることを

示します。適切なアクションを実行する必要がありますが、この際に注意して XMS スレッド化モデルのルールに従ってください (23 ページの『スレッド化モデル』を参照)。

## ConnectionFactory オブジェクトと Connection オブジェクト

ConnectionFactory オブジェクトには、アプリケーションが Connection オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、Connection オブジェクトを使用して Session オブジェクトを作成します。

ドット・ネットの場合、XMS アプリケーションは、最初に XMSFactoryFactory オブジェクトを使用して、必要なタイプのプロトコルに適した ConnectionFactory オブジェクトへの参照を取得します。この結果、この ConnectionFactory オブジェクトは、対象のプロトコル・タイプに対してのみ接続経路を作成できます。

XMS アプリケーションは、複数の接続経路を作成できます。また、マルチスレッド化アプリケーションは、複数のスレッドで 1 つの Connection オブジェクトを並行して使用できます。Connection オブジェクトは、アプリケーションとメッセージング・サーバー間の通信接続経路をカプセル化します。

接続経路は、以下のように複数の役割を果たします。

- アプリケーションが接続経路を作成した場合は、このアプリケーションを認証できます。
- アプリケーションは、固有のクライアント ID を接続経路に関連付けることができます。クライアント ID は、publish/subscribe・ドメインでの永続サブスクリプションをサポートするときに使用します。クライアント ID は次の 2 とおりの方法で設定できます。

接続のクライアント ID を割り当てる方法として望ましいのは、プロパティを使用してクライアント固有の ConnectionFactory オブジェクトでそれを構成し、作成する接続にそれを透過的に割り当てるという方法です。

クライアント ID を割り当てる別の方法は、Connection オブジェクトに対して設定されているプロバイダー固有の値を使用する方法です。その値によって、管理者により構成されている ID がオーバーライドされることはありません。それは、管理者によって指定された ID が存在しない場合のために提供されています。管理者の指定する ID が存在する場合、プロバイダー固有の値によってそれをオーバーライドしようとすると、例外がスローされます。アプリケーションが明示的に ID を設定する場合は、接続を作成した直後、かつその接続で他のアクションを実行する前に、ID を設定する必要があります。そうでないと、例外がスローされます。

XMS アプリケーションは、通常、1 つの接続経路、1 つ以上のセッション、およびいくつかのメッセージ・プロデューサーおよびメッセージ・コンシューマーを作成します。

接続経路の作成には通信接続経路の確立が必要であり、アプリケーションの認証が必要になる場合もあるため、システム・リソースの点で比較的成本が高くなります。

### 関連タスク

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### 関連資料

#### ICConnectionFactory (.NET インターフェイス用)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

#### ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

#### IDestination (.NET インターフェイス用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

#### Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## 接続の開始モードと停止モード

接続は、開始モードまたは停止モードで機能します。

アプリケーションが接続経路を作成したとき、この接続経路は停止モードになっています。接続経路が停止モードになっていると、アプリケーションは、セッションを初期設定してメッセージを送信することはできませんが、同期か非同期にかかわらず、受信することはできません。

アプリケーションは、`Start Connection` メソッドを呼び出すことによって接続を開始できます。接続経路が開始モードになっている場合、アプリケーションはメッセージの送信および受信を実行できます。その後、アプリケーションは `Stop Connection` メソッドおよび `Start Connection` メソッドを呼び出すことにより、接続の停止や再開を行うことができます。

### 関連概念

#### 接続の終了

アプリケーションは、`Close Connection` メソッドを呼び出すことにより、接続を終了します。

#### 例外処理

アプリケーションがメッセージを非同期に消費するためにのみ接続を使用する場合、アプリケーションが接続の問題を確認することができる唯一の方法は、例外リスナーを使用することです。

#### WebSphere サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

## 接続の終了

アプリケーションは、`Close Connection` メソッドを呼び出すことにより、接続を終了します。

アプリケーションが接続を終了すると、XMS は以下のアクションを実行します。

- 接続に関連するすべてのセッションを終了して、これらのセッションに関連する特定のオブジェクトを削除します。どのオブジェクトが削除されるかの詳細については、[40 ページの『オブジェクトの削除』](#)を参照してください。同時に、XMS は、セッション内で現在進行中のすべてのトランザクションをロールバックします。
- メッセージング・サーバーとの通信接続を終了します。
- 接続で使用されていたメモリーやその他の内部リソースを解放します。

XMS は、接続を終了する前に、セッション中に確認できなかったメッセージの受信確認は行いません。メッセージ受信の確認について詳しくは、[28 ページの『メッセージ肯定応答』](#)を参照してください。

### 関連概念

#### 接続の開始モードと停止モード

接続は、開始モードまたは停止モードで機能します。

#### 例外処理

アプリケーションがメッセージを非同期に消費するためにのみ接続を使用する場合、アプリケーションが接続の問題を確認することができる唯一の方法は、例外リスナーを使用することです。

#### WebSphere サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

## 例外処理

アプリケーションがメッセージを非同期に消費するためにのみ接続を使用する場合、アプリケーションが接続の問題を確認することができる唯一の方法は、例外リスナーを使用することです。

XMS ドット・ネット 例外は、すべて `System.Exception` から派生します。詳しくは、[50 ページの『ドット・ネットでのエラー処理』](#)を参照してください。

### 関連概念

#### 接続の開始モードと停止モード

接続は、開始モードまたは停止モードで機能します。

#### 接続の終了

アプリケーションは、Close Connection メソッドを呼び出すことにより、接続を終了します。

#### WebSphere サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

### **WebSphere サービス統合バスへの接続**

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

HTTP プロトコルは、TCP/IP 接続を直接行うことができない状態で使用できます。一般的な状態の1つは、2つの企業がメッセージを交換する場合など、ファイアウォールを介して通信する場合です。ファイアウォールを介して通信するために HTTP を使用することを、多くの場合、*HTTP* トンネリングと呼びます。ただし、HTTP トンネリングは、TCP/IP 接続を直接使用する場合よりも本質的に低速です。これは、HTTP ヘッダーがあると、転送されるデータの量が大幅に増加することや、HTTP プロトコルでは TCP/IP よりも多くの通信フローが必要になることが原因です。

TCP/IP 接続経路を作成するため、アプリケーションは、XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN プロパティが XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC に設定されている接続ファクトリーを使用できます。これはプロパティのデフォルト値です。接続経路が正常に作成されると、この接続経路の XMSC\_WPM\_CONNECTION\_PROTOCOL プロパティは XMSC\_WPM\_CP\_TCP に設定されます。

HTTP を使用する接続経路を作成するには、アプリケーションが使用する接続ファクトリーの XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN プロパティを、HTTP トランスポート・チャンネルを使用するよう構成されたインバウンド・トランスポート・チェーンの名前に設定する必要があります。接続経路が正常に作成されると、この接続経路の XMSC\_WPM\_CONNECTION\_PROTOCOL プロパティは、XMSC\_WPM\_CP\_HTTP に設定されます。トランスポート・チェーンの構成方法については、WebSphere Application Server の二重化の トランスポート・チェーン を参照してください。

アプリケーションには、ブートストラップ・サーバーに接続する場合、通信プロトコルについて同様の選択項目があります。接続ファクトリーの XMSC\_WPM\_PROVIDER\_ENDPOINTS プロパティは、ブートストラップ・サーバーの1つ以上のエンドポイント・アドレスの列です。各エンドポイント・アドレスのブートストラップ・トランスポート・チェーン・コンポーネントは、ブートストラップ・サーバーへの TCP/IP 接続の場合は XMSC\_WPM\_BOOTSTRAP\_TCP、HTTP を使用する接続の場合は XMSC\_WPM\_BOOTSTRAP\_HTTP になります。

#### **関連概念**

##### 接続の開始モードと停止モード

接続は、開始モードまたは停止モードで機能します。

##### 接続の終了

アプリケーションは、Close Connection メソッドを呼び出すことにより、接続を終了します。

##### 例外処理

アプリケーションがメッセージを非同期に消費するためにのみ接続を使用する場合、アプリケーションが接続の問題を確認することができる唯一の方法は、例外リスナーを使用することです。

#### **関連タスク**

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### **関連資料**

##### IConnectionFactory (.NET インターフェース用)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

##### ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

##### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

## Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## セッション数

Session は、メッセージを送受信する単一スレッド化されたコンテキストです。

アプリケーションはセッションを使用して、メッセージ、メッセージ・プロデューサー、メッセージ・コンシューマー、キュー・ブラウザー、および一時宛先を作成できます。また、アプリケーションはセッションを使用してローカル・トランザクションを実行することもできます。

アプリケーションは複数のセッションを作成できます。この場合、各セッションは、他のセッションとは関係なくメッセージを作成および消費します。個別のセッションまたは同一セッション内の 2 つのメッセージ・コンシューマーが同一トピックをサブスクライブする場合、これらのメッセージ・コンシューマーはそれぞれ、そのトピックについて公開されたメッセージのコピーを受信します。

Connection オブジェクトとは異なり、Session オブジェクトを複数のスレッドで並行して使用することはできません。Session オブジェクトの Close Session メソッドだけを、Session オブジェクトがその時点で使用しているスレッド以外のスレッドから呼び出すことができます。Close Session メソッドはセッションを終了し、そのセッションに割り振られていたシステム・リソースを解放します。

アプリケーションで複数のスレッドのメッセージを並行処理する必要がある場合、アプリケーションは、スレッドごとにセッションを作成し、そのセッションを、そのスレッド内の送信操作または受信操作に使用する必要があります。

## トランザクション化されたセッション

XMS アプリケーションは、ローカル・トランザクションを実行できます。ローカル・トランザクションとは、リソースに対する変更を伴うトランザクションのことです。対象となるリソースは、アプリケーションの接続先となるキュー・マネージャーまたはサービス統合バスのリソースです。

このトピック節の情報は、アプリケーションが IBM WebSphere MQ キュー・マネージャーまたは WebSphere サービス統合バスに接続する場合にのみ該当します。この情報は、ブローカーへのリアルタイム接続の場合は該当しません。

ローカル・トランザクションを実行するには、まず、セッションが処理されたことをパラメーターとして指定して Connection オブジェクトの Create Session メソッドを呼び出すことにより、アプリケーションでトランザクション化セッションを作成する必要があります。その後、そのセッション内で送受信されたすべてのメッセージは、トランザクションの順序でグループ化されます。トランザクションは、トランザクションが開始されてから送受信したメッセージがアプリケーションでコミットまたはロールバックされると終了します。

トランザクションをコミットするため、アプリケーションは Session オブジェクトの Commit メソッドを呼び出します。トランザクションがコミットされると、そのトランザクション内に送信されたすべてのメッセージは、他のアプリケーションに配信できるようになります。また、そのトランザクション内に受信したすべてのメッセージが認知されるので、メッセージング・サーバーはそれらのメッセージをアプリケーションへ再配信しなくなります。Point-to-Point ドメインの場合、メッセージング・サーバーは、受信したメッセージをサーバーのキューから除去する処理も行います。

トランザクションをロールバックするため、アプリケーションは Session オブジェクトの Rollback メソッドを呼び出します。トランザクションがロールバックされると、そのトランザクション内に送信されたすべてのメッセージはメッセージング・サーバーによって破棄されます。また、そのトランザクション内に受信したすべてのメッセージは再配信できるようになります。Point-to-Point ドメインでは、受信したメッセージがキューにプットバックされ、再び他のアプリケーションからも可視になります。

新規トランザクションは、アプリケーションがトランザクション化セッションを作成するか、Commit または Rollback メソッドを呼び出すと、自動的に開始します。したがって、トランザクション化されたセッションには常にアクティブなトランザクションが含まれます。

アプリケーションがトランザクション化されたセッションを閉じると、暗黙的なロールバックが行われます。アプリケーションが接続を閉じると、その接続のトランザクション化されたセッションすべてで暗黙的なロールバックが行われます。

トランザクションは、トランザクション化されたセッションに完全に包含されています。トランザクションがセッションをまたぐことはできません。つまり、アプリケーションは、トランザクション化された複数のセッションの中でメッセージを送受信したり、これらのすべてのアクションを単一のトランザクションとしてコミットまたはロールバックしたりすることはできません。

## 関連概念

### メッセージ肯定応答

未処理の各セッションには、アプリケーションが受信したメッセージの確認方法を決定する肯定応答モードがあります。3つの肯定応答モードが使用できます。肯定応答モードの選択は、アプリケーションの設計に影響を及ぼします。

### 非同期メッセージ配信

XMS は、1つのスレッドを使用して、あるセッションのすべての非同期メッセージ配信を処理します。このことは、一度に実行できるのは1つのメッセージ・リスナー関数または1つの `onMessage()` メソッドのみであるという意味です。

### 同期メッセージ配信

アプリケーションが `MessageConsumer` オブジェクトの `Receive` メソッドを使用している場合は、メッセージをアプリケーションに同期した状態で配信します。

### メッセージ送達モード

XMS は、次の2種類のメッセージ送達モードをサポートしています。

## メッセージ肯定応答

未処理の各セッションには、アプリケーションが受信したメッセージの確認方法を決定する肯定応答モードがあります。3つの肯定応答モードが使用できます。肯定応答モードの選択は、アプリケーションの設計に影響を及ぼします。

このトピック節の情報は、アプリケーションが IBM WebSphere MQ キュー・マネージャーまたは WebSphere サービス統合バスに接続する場合にのみ該当します。この情報は、ブローカーへのリアルタイム接続の場合は該当しません。

XMS が使用しているメッセージ受信を確認する仕組みは、JMS が使用している仕組みと同じです。

セッションが未処理である場合、アプリケーションが受信したメッセージを確認する方法は、セッションの肯定応答モードによって決まります。以下の段落で、3つの肯定応答モードについて説明します。

### XMSC\_AUTO\_ACKNOWLEDGE

セッションは、アプリケーションが受信した各メッセージを自動的に確認します。

メッセージがアプリケーションに同期化して配信されると、セッションは、`Receive` 呼び出しが正常に完了するたびにメッセージの受信を確認します。

アプリケーションはメッセージを正常に受信したが、障害のために肯定応答を実行できなかった場合、そのメッセージは再度配信できるようになります。このため、アプリケーションは再配信されるメッセージを処理する必要があります。

### XMSC\_DUPS\_OK\_ACKNOWLEDGE

セッションは、メッセージ選択時にアプリケーションが受信したメッセージを確認します。

この肯定応答モードを使用すると、セッションで行う必要のある処理量は減少しますが、メッセージ肯定応答を妨げる障害が発生すると、複数のメッセージが再配信の対象になることがあります。このため、アプリケーションは再配信される複数のメッセージを処理する必要があります。

### XMSC\_CLIENT\_ACKNOWLEDGE

`Message` クラスの `Acknowledge` メソッドを呼び出すことにより、受信したメッセージをアプリケーションが確認します。

アプリケーションは各メッセージの受信を個々に確認するか、または複数のメッセージを一括して受信し、受信した最後のメッセージに対してのみ `Acknowledge` メソッドを呼び出すことができます。

`Acknowledge` メソッドが呼び出されると、このメソッドの前の呼び出し以降に受信したすべてのメッセージが確認されます。

アプリケーションは、前述した肯定応答モードと関連して、`Session` クラスの `Recover` メソッドを呼び出すことにより、セッション中にメッセージの配信を停止することや再開することができます。以前に受信が

確認されなかったメッセージは、再配信されます。ただし、前回送達されたときと同じシーケンスで送達されるとは限りません。これらのメッセージが再送達されるまでの間に、より優先順位の高いメッセージが届いている可能性もありますし、オリジナルのメッセージの一部が有効期限切れになっている場合もあります。Point-to-Point ドメインの場合は、オリジナルのメッセージの一部が別のアプリケーションによって消費されている可能性もあります。

アプリケーションでは、メッセージの `JMSRedelivered` ヘッダー・フィールドの内容を調べることで、メッセージが再送達中かどうかを確認できます。アプリケーションでこの確認を行うには、`Message` クラスの `Get JMSRedelivered` メソッドを呼び出します。

## 関連概念

### トランザクション化されたセッション

XMS アプリケーションは、ローカル・トランザクションを実行できます。ローカル・トランザクションとは、リソースに対する変更を伴うトランザクションのことです。対象となるリソースは、アプリケーションの接続先となるキュー・マネージャーまたはサービス統合バスのリソースです。

### 非同期メッセージ配信

XMS は、1つのスレッドを使用して、あるセッションのすべての非同期メッセージ配信を処理します。このことは、一度に実行できるのは1つのメッセージ・リスナー関数または1つの `onMessage()` メソッドのみであるという意味です。

### 同期メッセージ配信

アプリケーションが `MessageConsumer` オブジェクトの `Receive` メソッドを使用している場合は、メッセージをアプリケーションに同期した状態で配信します。

### メッセージ送達モード

XMS は、次の2種類のメッセージ送達モードをサポートしています。

## 非同期メッセージ配信

XMS は、1つのスレッドを使用して、あるセッションのすべての非同期メッセージ配信を処理します。このことは、一度に実行できるのは1つのメッセージ・リスナー関数または1つの `onMessage()` メソッドのみであるという意味です。

あるセッションで複数のメッセージ・コンシューマーが複数のメッセージを非同期で受信しており、メッセージ・リスナー関数または `onMessage()` メソッドが1つのメッセージを1つのメッセージ・コンシューマーに配信している場合、同じメッセージを待っている他のメッセージ・コンシューマーは引き続き待機する必要があります。セッションへの配信を待機中のその他のメッセージも、引き続き待機する必要があります。

アプリケーションがメッセージを並行して配信する必要がある場合、XMS が複数のスレッドを使用して非同期メッセージ配信を処理するように、複数のセッションを作成します。このようにして、複数のメッセージ・リスナー関数または `onMessage()` メソッドを並行して実行できます。

コンシューマーにメッセージ・リスナーを割り当てても、セッションは非同期にはなりません。セッションが非同期になるのは、`Connection.Start` メソッドが呼び出されたときだけです。

`Connection.Start` メソッドが呼び出されるまでは、すべての同期呼び出しが可能です。

`Connection.Start` が呼び出されると、コンシューマーへのメッセージ配信が開始されます。

コンシューマーやプロデューサーの作成などの同期呼び出しを非同期セッションで行う必要がある場合は、`Connection.Stop` を呼び出す必要があります。`Connection.Start` メソッドを呼び出してメッセージの配信を開始することにより、セッションを再開できます。この唯一の例外が、メッセージをコールバック関数に配信するセッション・メッセージ配信スレッドです。このスレッドは、セッションのメッセージ・コールバック関数でどのような呼び出しでも(クローズ呼び出しを除く)行えます。

**注:** 非管理モードでは、WMQ.NET クライアントがコールバック関数内の `MQDISC` 呼び出しを使用することはできません。したがって、クライアント・アプリケーションが非同期受信モードの `MessageListener` コールバックでセッションを作成したり閉じたりすることはできません。`MessageListener` メソッドの外部でセッションを作成して処理します。

## 関連概念

### トランザクション化されたセッション

XMS アプリケーションは、ローカル・トランザクションを実行できます。ローカル・トランザクションとは、リソースに対する変更を伴うトランザクションのことです。対象となるリソースは、アプリケーションの接続先となるキュー・マネージャーまたはサービス統合バスのリソースです。

#### メッセージ肯定応答

未処理の各セッションには、アプリケーションが受信したメッセージの確認方法を決定する肯定応答モードがあります。3つの肯定応答モードが使用できます。肯定応答モードの選択は、アプリケーションの設計に影響を及ぼします。

#### 同期メッセージ配信

アプリケーションが MessageConsumer オブジェクトの Receive メソッドを使用している場合は、メッセージをアプリケーションに同期した状態で配信します。

#### メッセージ送達モード

XMS は、次の2種類のメッセージ送達モードをサポートしています。

### **同期メッセージ配信**

アプリケーションが MessageConsumer オブジェクトの Receive メソッドを使用している場合は、メッセージをアプリケーションに同期した状態で配信します。

Receive メソッドを使用すると、アプリケーションはメッセージを指定の期間または無期限に待機できません。あるいは、アプリケーションにメッセージを待機させない場合は、Receive with No Wait メソッドを使用できます。

#### **関連概念**

##### トランザクション化されたセッション

XMS アプリケーションは、ローカル・トランザクションを実行できます。ローカル・トランザクションとは、リソースに対する変更を伴うトランザクションのことです。対象となるリソースは、アプリケーションの接続先となるキュー・マネージャーまたはサービス統合バスのリソースです。

#### メッセージ肯定応答

未処理の各セッションには、アプリケーションが受信したメッセージの確認方法を決定する肯定応答モードがあります。3つの肯定応答モードが使用できます。肯定応答モードの選択は、アプリケーションの設計に影響を及ぼします。

#### 非同期メッセージ配信

XMS は、1つのスレッドを使用して、あるセッションのすべての非同期メッセージ配信を処理します。このことは、一度に実行できるのは1つのメッセージ・リスナー関数または1つの onMessage() メソッドのみであるという意味です。

#### メッセージ送達モード

XMS は、次の2種類のメッセージ送達モードをサポートしています。

### **メッセージ送達モード**

XMS は、次の2種類のメッセージ送達モードをサポートしています。

- 永続メッセージは、1回送信されます。メッセージング・サーバーは、メッセージのロギングなどの特殊な予防措置を取り、障害が発生した場合にも転送中に永続メッセージを失わないようにしています。
- 非永続メッセージが送信されるのは1回以内です。非永続メッセージは、障害が発生した場合、転送中に失われる可能性があるため、永続メッセージより信頼性は低くなります。

送達モードの選択は、信頼性とパフォーマンスとのトレードオフになります。非永続メッセージは、通常、永続メッセージより転送速度が高速になります。

#### **関連概念**

##### トランザクション化されたセッション

XMS アプリケーションは、ローカル・トランザクションを実行できます。ローカル・トランザクションとは、リソースに対する変更を伴うトランザクションのことです。対象となるリソースは、アプリケーションの接続先となるキュー・マネージャーまたはサービス統合バスのリソースです。

#### メッセージ肯定応答

未処理の各セッションには、アプリケーションが受信したメッセージの確認方法を決定する肯定応答モードがあります。3つの肯定応答モードが使用できます。肯定応答モードの選択は、アプリケーションの設計に影響を及ぼします。

#### 非同期メッセージ配信

XMS は、1つのスレッドを使用して、あるセッションのすべての非同期メッセージ配信を処理します。このことは、一度に実行できるのは1つのメッセージ・リスナー関数または1つの `onMessage()` メソッドのみであるという意味です。

#### 同期メッセージ配信

アプリケーションが `MessageConsumer` オブジェクトの `Receive` メソッドを使用している場合は、メッセージをアプリケーションに同期した状態で配信します。

## 宛先

XMS アプリケーションは、送信対象メッセージの宛先と受信対象メッセージの送信元を指定するときに `Destination` オブジェクトを使用します。

XMS アプリケーションは、`Destination` オブジェクトを実行時に作成することも、管理対象オブジェクトのリポジトリから事前定義の宛先を取得することもできます。

`ConnectionFactory` の場合と同様に、XMS アプリケーションで宛先を指定するための最も柔軟な方法は、宛先を管理対象オブジェクトとして定義する方法です。この方法を使用すると、C、C++、ドット・ネットの各言語で作成したアプリケーションと、Java で作成したアプリケーションが宛先の定義を共用できるようになります。管理対象の `Destination` オブジェクトのプロパティは、コードを変更せずに変更できます。

ドット・ネット アプリケーションの場合は、`CreateTopic` メソッドまたは `CreateQueue` メソッドを使用して宛先を作成します。これら2つのメソッドは、ドット・ネット API の `ISession` オブジェクトと `XMSFactoryFactory` オブジェクトの両方で使用できます。詳しくは、[48 ページの『.NET での宛先』](#) および [110 ページの『IDestination』](#) を参照してください。

## 関連資料

### [IDestination \(.NET インターフェース用\)](#)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### [Destination のプロパティ](#)

`Destination` オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティ (複数可) を指定することもできます。

トピックの URI は `topic://` というシーケンスで始まり、この後にトピック名が指定されます。また、他のトピック・プロパティを設定する名前と値のペアのリストを指定できます。トピック名を空にすることはできません。

この例を以下のドット・ネット コードのフラグメントに示します。

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

URI に使用できる名前と有効値など、トピックのプロパティの詳細については、[189 ページの『Destination のプロパティ』](#) を参照してください。

サブスクリプションで使用するトピック URI を指定するときに、ワイルドカードを使用できます。ワイルドカードの構文は、接続タイプとブローカーのバージョンによって異なります。以下のオプションが使用可能です。

- IBM WebSphere MQ V7.0 キュー・マネージャー (文字レベルのワイルドカード形式)
- IBM WebSphere MQ V7.0 キュー・マネージャー (トピック・レベルのワイルドカード形式)

- IBM WebSphere MQ V6.0 キュー・マネージャー (ブローカー V1 接続) (IBM WebSphere MQ V6.0 publish/subscribe)
- IBM WebSphere MQ V6.0 (ブローカー V2 接続またはブローカー V2 へのリアルタイム接続) (WebSphere Event Broker または WebSphere Message Broker)
- WebSphere サービス統合バス

## IBM WebSphere MQ V7.0 キュー・マネージャー (文字レベルのワイルドカード形式)

IBM WebSphere MQ V7.0 キュー・マネージャー (文字レベルのワイルドカード形式) では、以下のワイルドカード文字が使用されます。

- \* (0 以上の文字)
- 1 文字の ?
- % (エスケープ文字)

32 ページの表 1 に、このワイルドカード方式の使用法の例を示します。

URI	一致するトピック	例
"topic://Sport*Results"	「Sport」で始まり「Results」で終わるすべてのトピック	「topic://SportsResults」や「topic://Sport/Hockey/National/Div3/Results」
"topic://Sport?Results"	「Sport」、任意の 1 文字、「Result」という順のすべてのトピック	「topic://SportsResults」や「topic://SportXResults」
"topic://Sport/*ball*/Div?/Results*/???"	トピック	「topic://Sport/Football/Div1/Results/2002/Nov」や「topic://Sport/Netball/National/Div3/Results/02/Jan」

## IBM WebSphere MQ V7.0 キュー・マネージャー (トピック・レベルのワイルドカード形式)

IBM WebSphere MQ V7.0 キュー・マネージャー (トピック・レベルのワイルドカード形式) では、以下のワイルドカード文字が使用されます。

- # (複数レベルと一致)
- + (単一レベルと一致)

32 ページの表 2 に、このワイルドカード方式の使用法の例を示します。

URI	一致するトピック	例
"topic://Sport+/Results"	Sport と Results の間が一階層レベルであるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Ju-Jitsu/Results」
"topic://Sport#/Results"	「Sport/」で始まり「/Results」で終わるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Hockey/National/Div3/Results」
"topic://Sport/Football/#"	「Sport/Football/」で始まるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Football/TeamNews/Signings/Managerial」

## IBM WebSphere MQ V6.0 キュー・マネージャー (ブローカー V1 接続)

IBM WebSphere MQ V6.0 キュー・マネージャー (ブローカー V1 接続) では、以下のワイルドカード文字が使用されます。

- \* (0 以上の文字)
- 1 文字の ?
- % (エスケープ文字)

32 ページの表 1 に、このワイルドカード方式の使用法の例を示します。

## IBM WebSphere MQ V6.0 (ブローカー V2 接続またはブローカー V2 へのリアルタイム接続)

IBM WebSphere MQ V6.0 (ブローカー V2 接続またはブローカー V2 へのリアルタイム接続) では、以下のワイルドカード文字が使用されます。

- # (複数レベルと一致)
- + (単一レベルと一致)

32 ページの表 2 に、このワイルドカード方式の使用法の例を示します。

## WebSphere サービス統合バス

WebSphere サービス統合バス では、以下のワイルドカード文字が使用されます。

- \* (階層内の単一レベルの任意の文字に一致)
- // (0 以上のレベルに一致)
- //. 0 以上のレベル (トピック式の末尾) に一致させるため

33 ページの表 3 に、このワイルドカード方式の使用法の例を示します。

URI	一致するトピック	例
"topic://Sport/*ball/Results"	Sport と Results の間に、「ball」で終わる単一階層レベル名があるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Netball/Results」
"topic://Sport//Results"	「Sport/」で始まり「/Results」で終わるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Hockey/National/Div3/Results」
"topic://Sport/Football//."	「Sport/Football/」で始まるすべてのトピック	「topic://Sport/Football/Results」や「topic://Sport/Football/TeamNews/Signings/Managerial」
"topic://Sport/*ball//Results//."	トピック	「topic://Sport/Football/Results」や「topic://Sport/Netball/National/Div3/Results/2002/November」

### 関連概念

#### キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティ (複数可) を指定することもできます。

#### 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

#### 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

## 関連資料

### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティー

Destination オブジェクトのプロパティーの概要と、詳細な参照情報へのリンクを示します。

## キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティー (複数可) を指定することもできます。

キューの URI は `queue://` というシーケンスで始まり、この後にキュー名が続きます。また、他のキュー・プロパティーを設定する名前と値のペアのリストを指定できます。

IBM WebSphere MQ キュー (WebSphere Application Server のデフォルト・メッセージング・プロバイダー・キューを除く) の場合、キューがあるキュー・マネージャーをキューの前に指定できます。指定する場合は、キュー・マネージャー名とキュー名を `/` で区切ります。

キュー・マネージャーを指定する場合、このマネージャーはこのキューを使用する接続のために XMS が直接接続されているキュー・マネージャーであるか、またはこのキューからアクセスできるキュー・マネージャーでなければなりません。リモート・キュー・マネージャーは、キューからのメッセージの取り出し操作でのみサポートされており、キューへのメッセージの書き込み操作ではサポートされていません。詳しくは、IBM WebSphere MQ キュー・マネージャーの資料を参照してください。

キュー・マネージャーを指定しない場合は、追加の `/` 分離文字はオプションです。 `/` 文字の有無によってキューの定義が異なることはありません。

以下のキュー定義はすべて、XMS が直接接続されている `QM_A` というキュー・マネージャー上の `QB` という IBM WebSphere MQ キューに相当します。

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

## 関連概念

### トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティー (複数可) を指定することもできます。

### 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

### 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

## 関連資料

### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティー

Destination オブジェクトのプロパティーの概要と、詳細な参照情報へのリンクを示します。

## 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

一般にアプリケーションは、一時宛先を使用して要求メッセージに対する応答を受信します。要求メッセージに対する応答の送信先となる宛先を指定するため、アプリケーションは要求メッセージを表す Message オブジェクトの `Set JMSReplyTo` メソッドを呼び出します。呼び出しに宛先として一時宛先を指定できます。

セッションを使用して一時宛先が作成されますが、一時宛先の実際の有効範囲は、セッションの作成に使用された接続になります。接続のどのセッションでも、一時宛先のメッセージ・プロデューサーとメッセージ・コンシューマーを作成できます。一時宛先は、明示的に削除されるまで、あるいは接続が終了するまで存続します。

アプリケーションが一時キューを作成する場合、キューはアプリケーションの接続先メッセージング・サーバーに作成されます。アプリケーションがキュー・マネージャーに接続されている場合、**XMSC WMQ TEMPORARY MODEL** プロパティによって名前が指定されているモデル・キューから動的キューが作成され、動的キューの名前を形成するために使用される接頭部が **XMSC WMQ TEMP Q PREFIX** プロパティによって指定されます。アプリケーションがサービス統合バスに接続されている場合、一時キューがバス内に作成され、一時キューの名前を形成するために使用される接頭部が **XMSC WPM TEMP Q PREFIX** プロパティによって指定されます。

サービス統合バスに接続されているアプリケーションが一時トピックを作成する場合、一時トピックの名前を形成するために使用される接頭部は、**XMSC WPM TEMP TOPIC PREFIX** プロパティによって指定されます。

## 関連概念

### トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティ (複数可) を指定することもできます。

### キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティ (複数可) を指定することもできます。

### 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

## 関連資料

### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

スキームは、以下のとおりです。

接続のタイプ	ワイルドカード・スキーム	説明
IBM WebSphere MQ キュー・マネージャー	*	0 個以上の文字
	?	1 文字
	%	エスケープ文字
ブローカーへのリアルタイム接続	#	複数レベルと一致
	+	単一レベルと一致
WebSphere サービス統合バス	*	階層内の単一レベルの任意の文字に一致
	//	0 以上のレベルに一致
	//.	0 以上のレベルに一致 (Topic 式の終わり)

「WebSphere Application Server」資料の [トピック名とトピック式でのワイルドカード文字の使用](#) も参照してください。

## 関連概念

### トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティ (複数可) を指定することもできます。

### キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティ (複数可) を指定することもできます。

### 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

## 関連資料

### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## メッセージ・プロデューサー

XMS では、有効な宛先が指定された状態、または関連した宛先のない状態のいずれかで、メッセージ・プロデューサーを作成することができます。宛先のない状態でメッセージ・プロデューサーを作成する場合は、メッセージの送信時に有効な宛先を指定する必要があります。

### 関連した宛先のないメッセージ・プロデューサー

XMS .NET では、宛先のない状態でメッセージ・プロデューサーを作成できます。

.NET API の使用時に、宛先が関連付けられていないメッセージ・プロデューサーを作成するには、ISession オブジェクトの CreateProducer() メソッド (例えば、session.CreateProducer(null)) にパラメーターとして NULL を渡す必要があります。ただし、メッセージの送信時には有効な宛先を指定する必要があります。

### 関連した宛先のあるメッセージ・プロデューサー

このシナリオでは、メッセージ・プロデューサーを、有効な宛先を使用して作成します。送信操作の際に宛先を指定する必要はありません。

## メッセージ・コンシューマー

メッセージ・コンシューマーは、永続サブスクライバーと非永続サブスクライバー、および同期メッセージ・コンシューマーと非同期メッセージ・コンシューマーに分類することができます。

### 永続サブスクライバー

永続サブスクライバーは、特定のトピックに関してパブリッシュされたすべてのメッセージ (サブスクライバーが非アクティブだったときにパブリッシュされたメッセージも含む) を受信するメッセージ・コンシューマーです。

このトピック節の情報は、アプリケーションが IBM WebSphere MQ キュー・マネージャーまたは WebSphere サービス統合バスに接続する場合にのみ該当します。この情報は、ブローカーへのリアルタイム接続の場合は該当しません。

トピックの永続サブスクライバーを作成するため、アプリケーションは永続サブスクリプションを示す名前とトピックを表す Destination オブジェクトをパラメーターとして指定して、Session オブジェクトの Create Durable Subscriber メソッドを呼び出します。アプリケーションはメッセージ・セレクターを備えた永続サブスクライバーまたはメッセージ・セレクターがない永続サブスクライバーを作成できます。また、永続サブスクライバーがサブスクライバー自体の接続により公開されたメッセージを受信するかどうかを指定できます。

永続サブスクライバーの作成に使用されるセッションには、クライアント ID が関連付けられている必要があります。このクライアント ID は、セッション作成時に使用された接続に関連付けられていたクライアント ID と同一です。それは、24 ページの『[ConnectionFactories オブジェクトと Connection オブジェクト](#)』の記述に従って指定されます。

永続サブスクリプションを示す名前はクライアント ID において固有でなければなりません。したがって、クライアント ID は永続サブスクリプションの完全な固有 ID の一部を構成します。メッセージング・サーバーは永続サブスクリプションのレコードを維持しており、トピックについて公開されたすべてのメッセージが、永続サブスクライバーにより確認されるか、または有効期限が切れるまで保存されるようにします。

永続サブスクライバーが閉じた後も、メッセージング・サーバーは引き続き永続サブスクリプションのレコードを維持します。以前に作成された永続サブスクリプションを再利用するには、アプリケーションが同じサブスクリプション名を指定し、永続サブスクリプションに関連付けられていたものと同じクライアント ID のセッションを使用して、永続サブスクライバーを作成する必要があります。一度に1つのセッションだけが、特定の永続サブスクリプションの永続サブスクライバーを維持できます。

永続サブスクリプションの有効範囲は、サブスクリプションのレコードを維持するメッセージング・サーバーです。2つのアプリケーションがそれぞれ異なるメッセージング・サーバーに接続しており、各アプリケーションが同じサブスクリプション名とクライアント ID を使用して永続サブスクライバーを作成すると、完全に独立した2つの永続サブスクリプションが作成されます。

永続サブスクリプションを削除する場合、アプリケーションは永続サブスクリプションを示す名前をパラメーターとして指定し、Session オブジェクトの Unsubscribe メソッドを呼び出します。セッションに関連付けられているクライアント ID は、永続サブスクリプションに関連付けられているクライアント ID と同一でなければなりません。メッセージング・サーバーは保守している永続サブスクリプションのレコードを削除し、永続サブスクライバーにこれ以降メッセージを送信しなくなります。

既存のサブスクリプションを変更する場合、アプリケーションは同一サブスクリプション名とクライアント ID を使用し、異なるトピックまたはメッセージ・セクター (あるいはこの両方) を指定して永続サブスクライバーを作成できます。永続サブスクリプションの変更は、サブスクリプションを削除してから新規サブスクリプションを作成する操作と同等です。

IBM WebSphere MQ V7.0 キュー・マネージャーに接続するアプリケーションの場合は、XMS がサブスクライバー・キューを管理します。そのため、アプリケーションでは、サブスクライバー・キューを指定する必要はありません。指定すると、XMS は、サブスクライバー・キューを無視します。

ただし、IBM WebSphere MQ V6.0 キュー・マネージャーに接続するアプリケーションの場合は、永続サブスクライバーごとに、サブスクライバー・キューを指定しておく必要があります。トピックのサブスクライバー・キュー名を指定するには、トピックを表す Destination オブジェクトの [XMSC\\_WM\\_Q\\_DUR\\_SUBQ](#) プロパティを設定します。デフォルトのサブスクライバー・キューは SYSTEM.JMS.D.SUBSCRIBER.QUEUE です。

IBM WebSphere MQ V6.0 キュー・マネージャーに接続している複数の永続サブスクライバーで1つのサブスクライバー・キューを共有するか、永続サブスクライバーごとに専用のサブスクライバー・キューからメッセージを取り出すことができます。アプリケーションでどちらの方法を採用するかについては、「[IBM WebSphere MQ Java の使用](#)」を参照してください。

永続サブスクリプションのサブスクライバー・キューは変更できない点に注意してください。サブスクライバー・キューを変更する必要がある場合は、サブスクリプションを削除してから新規作成する方法のみ変更できます。

サービス統合バスに接続するアプリケーションの場合、各永続サブスクライバーには永続サブスクリプション・ホームが指定されている必要があります。同じ接続を使用するすべての永続サブスクライバーの永続サブスクリプション・ホームを指定するには、その接続の作成に使用される ConnectionFactory オブジェクトの [XMSC\\_WPM\\_DUR\\_SUB\\_HOME](#) プロパティを設定します。個別のトピックの永続サブスクリプション・ホームを指定するには、トピックを表す Destination オブジェクトの [XMSC\\_WPM\\_DUR\\_SUB\\_HOME](#) プロパティを設定します。接続を使用する永続サブスクライバーをアプリケーションが作成する前に、その接続の永続サブスクリプション・ホームを指定する必要があります。宛先に対して指定されている値は、接続に対して指定されている値に優先します。

## 非永続サブスクライバー

非永続サブスクライバーは、サブスクライバーがアクティブである間に公開されたメッセージのみを受信するメッセージ・コンシューマーです。サブスクライバーがアクティブではないときに配信されたメッセージは失われます。

このトピック節の情報は、IBM WebSphere MQ V6.0 キュー・マネージャーで publish/subscribe メッセージングを使用している場合にのみ関係します。

接続の終了前または終了中にコンシューマー・オブジェクトが削除されていない場合は、アクティブでないサブスクライバーのブローカー・キューにメッセージが残ることがあります。

このような場合は、IBM WebSphere MQ Classes for JMS のクリーンアップ・ユーティリティを使用してキューからこれらのメッセージを消去できます。このユーティリティの使用の詳細については、「*IBM WebSphere MQ Java の使用*」を参照してください。また、このキューに多数のメッセージが残る場合は、サブスクライバー・キューのキュー項目数を増やす必要があります。

## 同期メッセージ・コンシューマー

同期メッセージ・コンシューマーは、キューから同期でメッセージを受信します。

同期メッセージ・コンシューマーは、メッセージを1つずつ受信します。Receive (待機間隔) メソッドが使用された場合、呼び出しは、指定された期間(ミリ秒単位)のみメッセージを待機するか、メッセージ・コンシューマーがクローズするまで待機します。

ReceiveNoWait() メソッドが使用された場合、同期メッセージ・コンシューマーは、遅延なしでメッセージを受信します。次のメッセージが使用可能になると、直ちにそのメッセージを受信します。それ以外の場合は、ヌルの Message オブジェクトへのポインターが返されます。

## 非同期メッセージ・コンシューマー

非同期メッセージ・コンシューマーは、キューから非同期でメッセージを受信します。キューで新しいメッセージが使用可能になるたびに、アプリケーションによって登録されたメッセージ・リスナーが呼び出されます。

## 有害メッセージ

有害メッセージとは、受信側の MDB アプリケーションによって処理できないメッセージです。有害メッセージが見つかったら、XMS MessageConsumer オブジェクトは、BOQNAME および BOTHRESH の2つのキュー・プロパティに従って、そのメッセージを再キューイングできます。

環境によっては、MDB に配信されたメッセージが IBM WebSphere MQ キューにロールバックされることがあります。これは、例えば、メッセージが1つの作業単位の中で配信され、その後、その作業単位がロールバックされた場合に発生する可能性があります。通常、ロールバックされたメッセージは再配信されますが、メッセージのフォーマットが正しくないと、MDB が繰り返し失敗し、したがってメッセージを配信できなくなります。こうしたメッセージは有害メッセージと呼ばれます。こうした有害メッセージを後で調査するために別のキューに自動的に転送したり、廃棄したりするように、IBM WebSphere MQ を構成することができます。このように IBM WebSphere MQ を構成する方法については、[ASF での有害メッセージの処理](#)を参照してください。

時折、不適切なフォーマットのメッセージがキューに到着する場合があります。ここで言う「不適切なフォーマット」とは、受信側のアプリケーションがメッセージを正しく処理できないことを意味します。このようなメッセージがあると、受信側のアプリケーションに障害が発生したり、この不適切なフォーマットのメッセージがアプリケーションによってバックアウトされたりすることがあります。そして、メッセージは繰り返し入力キューに送達され、アプリケーションによって繰り返しバックアウトされる可能性があります。これらのメッセージを、有害メッセージと呼びます。XMS MessageConsumer オブジェクトは、有害メッセージを検出して、代替宛先にそれらを再転送します。

IBM WebSphere MQ キュー・マネージャーは、それぞれのメッセージがバックアウトされた回数を記録しています。この回数が、構成可能なしきい値に達すると、メッセージ・コンシューマーはそのメッセージを名前付きのバックアウト・キューに再キューイングします。この再キューイングが何らかの理由により失敗すると、メッセージは入力キューから除去され、送達不能キューに再キューイングされるか、または廃棄されます。

XMS ConnectionConsumer オブジェクトは、同じ方法で、同じキュー・プロパティを使用して、有害メッセージを処理します。複数の接続コンシューマーが同じキューをモニターしている場合は、リキューが

行われるしきい値の回数を超えても有害メッセージがアプリケーションに送達されることがあります。この動作の原因は、接続コンシューマーが個別にキューをモニターして有害メッセージを再キューイングする方法にあります。

しきい値およびバックアウト・キューの名前は、IBM WebSphere MQ キューの属性です。これらの属性の名前は、BackoutThreshold および BackoutRequeueQName です。これらの属性が適用されるキューは、以下のとおりです。

- Point-to-Point メッセージングの場合、これは基礎ローカル・キューです。これは、メッセージ・コンシューマーおよび接続コンシューマーがキューの別名を使用する場合に重要です。
- IBM WebSphere MQ メッセージング・プロバイダーの通常モードにおけるパブリッシュ/サブスクライブ・メッセージングの場合、これは、Topic の管理キューの作成元であるモデル・キューです。
- IBM WebSphere MQ メッセージング・プロバイダーのマイグレーション・モードにおけるパブリッシュ/サブスクライブ・メッセージングの場合、これは、TopicConnectionFactory オブジェクトで定義された CCSUB キュー、または Topic オブジェクトで定義された CCDSUB キューです。

BackoutThreshold 属性および BackoutRequeueQName 属性を設定するには、次の MQSC コマンドを実行します。

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

パブリッシュ/サブスクライブ・メッセージングでは、システムがサブスクリプションごとに動的キューを作成する場合、これらの属性値は IBM WebSphere MQ classes for JMS モデル・キュー SYSTEM.JMS.MODEL.QUEUE。これらの設定を変更するには、以下を使用できます。

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

バックアウトのしきい値がゼロの場合、ポイズン・メッセージの処理は不可となり、ポイズン・メッセージは入力キュー上に残ります。それ以外の場合は、バックアウト・カウントがしきい値に達すると、メッセージは指定されたバックアウト・キューに送信されます。バックアウト・カウントがしきい値に達してもメッセージをバックアウト・キューに入れることができないときは、メッセージは送達不能キューに送信されるか、廃棄されます。この状況は、バックアウト・キューが定義されていない場合、または MessageConsumer オブジェクトがメッセージをバックアウト・キューに送信できない場合に発生します。

#### ASF での有害メッセージの処理

Application Server Facilities (ASF) を使用する場合は、MessageConsumer の代わりに ConnectionConsumer で有害メッセージを処理します。ConnectionConsumer は、キューの BackoutThreshold および BackoutRequeueQName プロパティーに従ってメッセージをリキューします。

アプリケーションが ConnectionConsumers を使用している場合、メッセージがバックアウトされる状況は、アプリケーション・サーバーが提供するセッションによって異なります。

- セッションが非トランザクション化セッションで、AUTO\_ACKNOWLEDGE または DUPS\_OK\_ACKNOWLEDGE が指定されている場合、メッセージがバックアウトされるのは、システム・エラーの後、またはアプリケーションが予期せずに終了した場合のみです。
- セッションが非トランザクション化セッションで、CLIENT\_ACKNOWLEDGE が指定されている場合、無応答メッセージは、アプリケーション・サーバーが Session.recover() を呼び出すことによってバックアウトすることができます。

一般に、MessageListener のクライアント実装またはアプリケーション・サーバーが Message.acknowledge() を呼び出します。Message.acknowledge() は、これまでそのセッションで配信されたすべてのメッセージに応答します。

- セッションがトランザクション化セッションの場合、無応答メッセージは、アプリケーション・サーバーが Session.rollback() を呼び出すことによってバックアウトすることができます。

## キュー・ブラウザー

アプリケーションは、キュー・ブラウザーを使用して、キュー上のメッセージを参照します。その際にメッセージは除去されません。

キュー・ブラウザーを作成するには、アプリケーションは、ISession オブジェクトの Create Queue Browser メソッドを呼び出し、参照するキューを示す Destination オブジェクトをパラメーターとして指定します。アプリケーションはメッセージ・セレクターを備えたキュー・ブラウザーまたはメッセージ・セレクターのないキュー・ブラウザーを作成できます。

キュー・ブラウザーの作成後、アプリケーションは、IQueueBrowser オブジェクトの GetEnumerator メソッドを呼び出して、キューにあるメッセージのリストを取得できます。このメソッドは、Message オブジェクトのリストをカプセル化する列挙子を戻します。リスト内の Message オブジェクトの順序は、メッセージがキューから取り出される順序と同じです。アプリケーションは列挙子を使用して、各メッセージを順番に参照できます。

メッセージがキューに書き込まれたり削除されたりすると、列挙子は動的に更新されます。アプリケーションが IEnumerator.MoveNext() を呼び出してキュー上の次のメッセージを参照するたびに、そのメッセージにはキューの現在の内容が反映されます。

アプリケーションは、特定のキュー・ブラウザーに対して GetEnumerator メソッドを複数回呼び出すことができます。呼び出しごとに、新しい列挙子が返されます。したがって、アプリケーションは複数の列挙子を使用してキューのメッセージを参照し、キュー内の複数の位置を維持することができます。

アプリケーションはキュー・ブラウザーを使用して、キューから除去する適切なメッセージを検索し、メッセージ・セレクターを備えたメッセージ・コンシューマーを使用してメッセージを除去することができます。メッセージ・セレクターは、JMSMessageID ヘッダー・フィールドの値に基づいてメッセージを選択できます。このフィールドをはじめとする JMS メッセージ・ヘッダーのフィールドについては、[72 ページの『XMS メッセージのヘッダー・フィールド』](#)を参照してください。

## リクエスター

アプリケーションはリクエスターを使用して要求メッセージを送信し、応答を待機して受信します。

多くのメッセージング・アプリケーションは、要求メッセージを送信して応答を待機するアルゴリズムに基づいています。XMS の Requestor というクラスは、このようなスタイルのアプリケーションを開発するときに役立ちます。

リクエスターを作成するため、アプリケーションは Requestor クラスの Create Requestor コンストラクターを呼び出します。このとき、Session オブジェクトと、要求メッセージの送信先を示す Destination オブジェクトがパラメーターとして指定されます。セッションがトランザクション化されてはなりません。また、セッションには肯定応答モード XMSC\_CLIENT\_ACKNOWLEDGE は設定できません。コンストラクターは、応答メッセージの送信先となる一時キューまたは一時トピックを自動的に作成します。

アプリケーションは、リクエスターの作成後に、Requestor オブジェクトの Request メソッドを呼び出して要求メッセージを送信し、要求メッセージを受信したアプリケーションからの応答を待機して受信することができます。応答の受信またはセッションの終了のいずれかが発生するまで、呼び出しは待機します。リクエスターが必要とする応答は、要求メッセージごとに1つのみです。

アプリケーションがリクエスターを閉じると、一時キューまたは一時トピックは削除されます。ただし関連付けられているセッションは閉じません。

## オブジェクトの削除

アプリケーションが、作成した XMS オブジェクトを削除すると、XMS は、このオブジェクトに割り振られていた内部リソースを解放します。

アプリケーションが XMS オブジェクトを作成すると、XMS は、このオブジェクトにメモリーとその他の内部リソースを割り振ります。XMS は、XMS が内部リソースを解放した時点で、アプリケーションがオブジェクトの close メソッドまたは delete メソッドを呼び出して、明示的にオブジェクトを削除するまで、内部リソースを保存します。アプリケーションが、既に削除されているオブジェクトを削除しようとする、呼び出しは無視されます。

アプリケーションが Connection オブジェクトまたは Session オブジェクトを削除すると、XMS は特定の関連オブジェクトを自動的に削除し、その内部リソースを解放します。これらは Connection オブジェクトまたは Session オブジェクトによって作成されたオブジェクトで、これらのオブジェクトから独立した機能を持っていません。これらのオブジェクトを 41 ページの表 4 に示します。

注：アプリケーションが従属セッションとの接続を終了すると、これらのセッションに従属するすべてのオブジェクトも削除されます。従属オブジェクトがあるのは、Connection オブジェクトまたは Session オブジェクトのみです。

削除されたオブジェクト	メソッド	自動的に削除される従属オブジェクト
接続	Close Connection	ConnectionMetaData オブジェクトおよび Session オブジェクト
Session	Close Session	MessageConsumer、MessageProducer、QueueBrowser、Requestor の各オブジェクト

## XMS による管理 IBM WebSphere MQ XA トランザクション

XMS で管理 IBM WebSphere MQ XA トランザクションを使用できます。

XMS で XA トランザクションを使用するには、トランザクション化セッションを作成する必要があります。XA トランザクションを使用する場合は、XMS セッションではなく、分散トランザクション・コーディネーター (DTC) のグローバル・トランザクションによって、トランザクションを制御します。XA トランザクションを使用するときに、XMS セッションで Session.commit または Session.rollback を実行することはできません。その代わりに、DTC メソッド Transscope.Commit または Transscope.Rollback を使用して、トランザクションのコミットまたはロールバックを実行します。XA トランザクションのためにセッションを使用している場合、そのセッションを使用して作成するプロデューサーまたはコンシューマーは、その XA トランザクションの一部でなければなりません。そのプロデューサーまたはコンシューマーを XA トランザクションの範囲外の操作のために使用することはできません。つまり、XA トランザクションの外で Producer.send や Consumer.receive のような操作を実行するために使用することはできないということです。

以下の場合、IllegalStateException 例外オブジェクトがスローされます。

- XA トランザクション化セッションを Session.commit または Session.rollback のために使用する場合。
- XA トランザクション化セッションで使用したことがあるプロデューサー・オブジェクトまたはコンシューマー・オブジェクトを XA トランザクションの範囲外で使用する場合。

非同期コンシューマーでは、XA トランザクションはサポートされていません。

注：

1. XA トランザクションがコミットされる前に、Producer、Consumer、Session、Connection のいずれかのオブジェクトを閉じる操作が実行される場合もあります。そのような場合は、トランザクションに含まれているメッセージがロールバックされます。さらに、XA トランザクションがコミットされる前に接続で障害が発生した場合も、トランザクションに含まれているすべてのメッセージがロールバックされます。Producer オブジェクトの場合のロールバックは、メッセージがキューに書き込まれないという意味です。Consumer オブジェクトの場合のロールバックは、メッセージがキューに残ったままになるという意味です。
2. Producer オブジェクトが TimeToLive を指定してメッセージを TransactionScope に入れた場合は、その存続時間の経過後に commit を実行しても、commit の実行前にメッセージの有効期限が切れた状態になります。その場合、Consumer オブジェクトがそのメッセージを受け取ることはできません。
3. Session オブジェクトを複数のスレッドで使用することはサポートされていません。複数のスレッドで Session オブジェクトを共用するトランザクションの使用もサポートされていません。

## XMS プリミティブ型

XMS には、Java の 8 個のプリミティブ型 (byte、short、int、long、float、double、char、および boolean) に相当するデータ型が用意されています。これにより、XMS と JMS の間で、データの損失や破損が発生することなくメッセージを交換することができます。

42 ページの表 5 に、XMS の各プリミティブ型に対応する Java のデータ型、サイズ、最小値、および最大値を示します。

XMS データ型	互換性のある Java データ型	サイズ	最小値	最大値:
System.Boolean	boolean	32 ビット	false	true
System.SBYTE	byte	8 ビット	$-2^7$ (-128)	$2^7-1$ (127)
System.BYTE	byte	8 ビット	$-2^7$ (-128)	$2^7-1$ (127)
System.CHAR	byte	8 ビット	$-2^7$ (-128)	$2^7-1$ (127)
System.Int16	不足	16 ビット	$-2^{15}$ (-32768)	$2^{15}-1$ (32767)
System.Int32	int	32 ビット	$-2^{31}$ (-2147483648)	$2^{31}-1$ (2147483647)
System.Int64	long	64 ビット	$-2^{63}$ (-9223372036854775808)	$2^{63}-1$ (9223372036854775807)
System.Single	float	32 ビット	-3.402823E-38 (精度 7 桁)	3.402823E+38 (精度 7 桁)
System.Double	double	64 ビット	-1.79769313486231E-308 (精度 15 桁)	1.79769313486231E+308 (精度 15 桁)

### 関連概念

#### オブジェクトの属性とプロパティ

XMS オブジェクトには、オブジェクトの特性である属性とプロパティを設定できます。これらはさまざまな方法で実装されます。

#### プロパティ値のデータ型の暗黙的な変換

アプリケーションがプロパティの値を取得するときに、XMS によりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMS がその変換を実行する方法を規定します。

### 関連資料

#### アプリケーション・データの要素のデータ型

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できる必要があります。

## プロパティ値のデータ型の暗黙的な変換

アプリケーションがプロパティの値を取得するときに、XMS によりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMS がその変換を実行する方法を規定します。

オブジェクトのプロパティには名前と値が指定されていて、その値にはデータ型が関連付けられています。プロパティの値は、プロパティ・タイプとも呼ばれます。

アプリケーションは、PropertyContext クラスのメソッドを使用してオブジェクトのプロパティを取得および設定します。プロパティ値を取得するため、アプリケーションは一般にプロパティ・タイプに対応したメソッドを呼び出します。例えば、整数プロパティの値を取得するには、アプリケーションは一般に GetIntProperty メソッドを呼び出します。

ただし、アプリケーションがプロパティの値を取得する際に、XMSによりこの値のデータ型を別のデータ型に変換できます。例えば、整数プロパティの値を取得するには、アプリケーションは GetStringProperty メソッドを呼び出すことで、そのプロパティの値を文字列として取得することができます。XMS でサポートされている変換を 43 ページの表 6 に示します。

表 6. サポートされるプロパティ値のデータ型の変換	
プロパティ・タイプ	サポートされているターゲット・データ・タイプ
System.String	System.Boolean, System.Double, System.Float, System.Int32, System.Int64, System.SByte, System.Int16
System.Boolean	System.String, System.SByte, System.Int32, System.Int64, System.Int16
System.Char	System.String
System.Double	System.String
System.Float	System.String, System.Double
System.Int32	System.String, System.Int64
System.Int64	System.String
System.SByte	System.String, System.Int32, System.Int64, System.Int16
System.SByte array	System.String
System.Int16	System.String, System.Int32, System.Int64

以下の汎用ルールは、サポートされる変換を規定します。

- 変換中にデータが失われない限り、数値プロパティ値のデータ型を変換できます。例えば、データ型 System.Int32 のプロパティの値を、データ型 System.Int64 の値に変換することはできますが、データ型 System.Int16 の値に変換することはできません。
- どのデータ・タイプのプロパティ値でも、文字列に変換できる。
- 文字列・プロパティ値は、文字列が変換用に正しくフォーマットされていれば、任意の他のデータ・タイプに変換できます。正しい形式になっていない文字列・プロパティ値をアプリケーションが変換しようとする、XMS からエラーが戻される場合があります。
- アプリケーションがサポートされていない変換を実行しようとする、XMS からエラーが戻されます。

プロパティ値のデータ型が変換される時、以下のルールが適用されます。

- ブール型プロパティの値を文字列に変換する場合、値 true は文字列 "true" に変換され、値 false は文字列 "false" に変換されます。
- ブール・プロパティ値を数値データ型 (System.SByte を含む) に変換すると、値 true は 1 に変換され、値 false は 0 に変換されます。
- 文字列・プロパティ値をブール値に変換する場合、文字列 "true" (大文字と小文字を区別しない) または "1" は true に変換され、文字列 "false" (大文字と小文字を区別しない) または "0" は false に変換されます。その他の文字列はいずれも変換できません。
- 文字列・プロパティ値を、データ型 System.Int32、System.Int64、System.SByte、または System.Int16 の値に変換する場合、変換する文字列は以下の形式でなければなりません。

*[blanks][sign]digits*

文字列・コンポーネントは、以下のように定義されます。

**blanks**

オプションの先行空白文字。

**sign**

オプションの正符号 (+) または負符号 (-) 文字。

**digits**

数字 (0 から 9 まで) の連続シーケンス。少なくとも 1 つの数字が存在している必要があります。

数字のシーケンスの後のストリングには数字以外の文字を含めることができますが、それらの文字の最初のものに達するとすぐに変換は停止します。ストリングは 10 進整数を表すと想定されます。

ストリングの形式が正しくないと、XMS からエラーが戻されます。

- ストリング・プロパティ値をデータ型 System.Double または System.Float の値に変換する場合、変換するストリングは以下の形式でなければなりません。

`[blanks][sign][digits][point[d_digits]][e_char[e_sign]e_digits]`

ストリング・コンポーネントは、以下のように定義されます。

**blanks**

(オプション) 先行空白文字。

**sign**

(オプション) 正符号 (+) または負符号 (-) 文字。

**digits**

数字 (0 から 9 まで) の連続シーケンス。 `digits` か `d_digits` のいずれかに、少なくとも 1 つの数字が必要です。

**point**

(オプション) 小数点 (.)。

**d\_digits**

数字 (0 から 9 まで) の連続シーケンス。 `digits` か `d_digits` のいずれかに、少なくとも 1 つの数字が必要です。

**e\_char**

指数文字。 `E` または `e` のいずれかです。

**e\_sign**

(オプション) 指数の正符号 (+) または負符号 (-) 文字。

**e\_digits**

指数用の、数字 (0-9) の連続シーケンス。ストリングに指数文字がある場合、1 つ以上の数字がなければなりません。

数字のシーケンスの後、または指数を表すオプションの文字の後のストリングには数字以外の文字を含めることができますが、それらの文字の最初のものに達するとすぐに変換は停止します。ストリングは、10 の累乗の指数を持つ 10 進浮動小数点数を表すと想定されます。

ストリングの形式が正しくないと、XMS からエラーが戻されます。

- 数値プロパティ値 (データ型 System.SByte のプロパティ値を含む) をストリングに変換する場合、値は、その値に対応する ASCII 文字を含むストリングではなく、その値を 10 進数として表現するストリングに変換されます。例えば、整数 65 はストリング "65" に変換され、ストリング "A" に変換されるわけではありません。
- バイト配列プロパティ値をストリングに変換する場合、各バイトはバイトを表す 2 つの 16 進文字に変換されます。例えば、バイト配列 {0xF1, 0x12, 0x00, 0xFF} がストリング "F11200FF" に変換されます。

プロパティ・タイプのデータ型変換は、Property クラスと PropertyContext クラスの両方のメソッドによりサポートされています。

**関連概念**

オブジェクトの属性とプロパティ

XMS オブジェクトには、オブジェクトの特性である属性とプロパティを設定できます。これらはさまざまな方法で実装されます。

XMS プリミティブ型

XMS には、Java の 8 個のプリミティブ型 (byte、short、int、long、float、double、char、および boolean) に相当するデータ型が用意されています。これにより、XMS と JMS の間で、データの損失や破損が発生することなくメッセージを交換することができます。

**関連資料**

マップ・メッセージ

マップ・メッセージの本文には、名前値のペアが1組含まれており、それぞれの値には関連付けられたデータ型があります。

#### ストリーム・メッセージ

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

#### アプリケーション・データの要素のデータ型

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できる必要があります。

## イテレーター

イテレーターは、オブジェクトのリストとこのリストの現在位置を維持するカーソルをカプセル化します。イテレーターの概念は、Message Service Client for C/C++ の場合と同じように、Message Service Client for .NET の IEnumerator インターフェースを使用することで実装されます。

イテレーターを作成すると、カーソルの位置は最初のオブジェクトの前になります。アプリケーションは、イテレーターを使用して各オブジェクトを順番に取得します。

Message Service Client for C/C++ の Iterator クラスは、Java の Enumerator クラスに相当します。XMS ドット・ネットは Java に類似し、IEnumerator インターフェースを使用します。

アプリケーションは、IEnumerator を使用して次のタスクを実行できます。

- メッセージのプロパティを取得する
- マップ・メッセージの本文で名前と値の対を取得する
- キューに入っているメッセージを参照する
- 接続でサポートされている JMS 定義のメッセージ・プロパティの名前を取得する

## コード化文字セット ID

XMS .NET では、ストリングはすべて、ネイティブの .NET ストリングを使用して渡されます。この場合のエンコード方式は固定であるため、解釈するためにこれ以上の情報は不要です。そのため、XMS .NET アプリケーションに XMSC\_CLIENT\_CCSD プロパティは必要ありません。

## XMS エラーおよび例外コード

XMS では、障害を示す一定の範囲のエラー・コードが使用されています。エラー・コードはリリースごとに異なる可能性があるため、本書では明示的には示していません。本書で示されているのは XMS 例外コード (XMS\_X\_... の形式) だけです。これは、このコードが XMS のリリース間で同一であるためです。

## ユーザー独自のアプリケーションの作成

ユーザー独自のアプリケーションをビルドする方法は、サンプル・アプリケーションをビルドする場合と同様です。

21 ページの『ドット・ネット サンプル・アプリケーションの作成』トピック節の説明に従って、ドット・ネット アプリケーションを作成します。ここでは、ユーザー独自のドット・ネット アプリケーションを作成するために必要な前提条件も示されています。ユーザー独自のアプリケーションの作成方法についての補足指示については、各サンプル・アプリケーションの Make ファイルを活用してください。

**ヒント:** 障害発生時の問題診断を支援するため、シンボルを組み込んだアプリケーションをコンパイルすると便利な場合があります。

### 関連資料

#### ドット・ネット インターフェース

このトピック節では、ドット・ネット クラスのインターフェースとそのプロパティおよびメソッドについて説明します。

#### XMS オブジェクトのプロパティ

この節章では、XMS で定義したオブジェクトのプロパティについて説明します。

## XMS による自動 IBM WebSphere MQ クライアント再接続

IBM WebSphere MQ V7.1 以上のクライアントをメッセージング・プロバイダーとして使用している場合は、ネットワーク、キュー・マネージャー、サーバーで障害が発生した後に、自動的に再接続するように XMS クライアントを構成できます。

MQConnectionFactory クラスの WMQ\_CONNECTION\_NAME\_LIST プロパティと WMQ\_CLIENT\_RECONNECT\_OPTIONS プロパティを使用して、クライアント接続が自動的に再接続するように構成できます。自動クライアント再接続では、接続で障害が発生した後にクライアントが再接続します。あるいは、オプションとしてキュー・マネージャーの停止後に自動クライアント再接続を利用することもできます。クライアント・アプリケーションの設計によっては、自動再接続に適さないクライアント・アプリケーションもあります。

自動再接続が可能なクライアント接続は、接続が確立した時点で再接続が可能な状態になります。

注: Client Reconnect Options、Client Reconnect Timeout、Connection Namelist の各プロパティは、クライアント・チャンネル定義テーブル (CCDT) で設定することもできます。または、mqclient.ini ファイルでクライアント再接続を有効にすることによって設定することも可能です。

注: ConnectionFactory オブジェクトと CCDT の両方で再接続プロパティを設定した場合の優先順位に関する規則は、以下のようになります。ConnectionFactory オブジェクトで接続名リスト・プロパティのデフォルト値が設定されている場合は、CCDT が優先されます。接続名リストがデフォルト値に設定されていない場合は、ConnectionFactory オブジェクトで設定されているプロパティ値が優先されます。接続名リストのデフォルト値は localhost(1414) です。

## XMS .NET アプリケーションの作成

この節章では、XMS ドット・ネット アプリケーションを作成する場合に役立つ情報を提供します。

この節章では、XMS ドット・ネット アプリケーションを作成する場合に固有の情報を提供します。XMS アプリケーションを作成する場合の一般情報については、[22 ページの『XMS アプリケーションの書き込み』](#)を参照してください。

節章には、以下のトピック節が含まれています。

- [46 ページの『ドット・ネットのデータ型』](#)
- [47 ページの『ドット・ネットにおける管理操作および非管理操作』](#)
- [48 ページの『.NET での宛先』](#)
- [48 ページの『ドット・ネットでのプロパティ』](#)
- [49 ページの『ドット・ネットでの存在しないプロパティの処理』](#)
- [50 ページの『ドット・ネットでのエラー処理』](#)
- [50 ページの『ドット・ネットでのメッセージ・リスナーおよび例外リスナー』](#)

### 関連資料

[ドット・ネットインターフェース](#)

このトピック節では、ドット・ネット クラスのインターフェースとそのプロパティおよびメソッドについて説明します。

## ドット・ネットのデータ型

XMS .NET は、System.Boolean、System.Byte、System.SByte、System.Char、System.String、System.Single、System.Double、System.Decimal、System.Int16、System.Int32、System.Int64、System.UInt16、System.UInt32、System.UInt64、および System.Object をサポートしています。XMS .NET のデータ型は、XMS C++ のデータ型とは異なります。この節章の情報を使用すれば、対応するデータ型を特定できます。

次の表に、対応する XMS .NET と XMS C++ のデータ型を示し、簡単に説明します。

表 7. XMS.NET と XMS C++ のデータ型

XMS .NET のデータ型	XMS C++ のデータ型	説明
System.SByte	xmsSBYTE xmsINT8	符号付き 8 ビット値
System.Byte	xmsBYTE xmsUINT8	符号なし 8 ビット値
System.Int16	xmsINT16 xmsSHORT	符号付き 16 ビット値
System.UInt16	xmsUINT16 xmsUSHORT	符号なし 16 ビット値
System.Int32	xmsINT32 xmsINT	符号付き 32 ビット値
System.UInt32	xmsUINT32 xmsUINT	符号なし 32 ビット値
System.Int64	xmsLONG xmsINT64	符号付き 64 ビット値
System.UInt64	xmsULONG xmsUINT64	符号なし 64 ビット値
System.Char	xmsCHAR16	符号なし 16 ビット文字 (.NET の場合は Unicode)
System.Single	xmsFLOAT	IEEE 32 ビット浮動小数点
System.Double	xmsDOUBLE	IEEE 64 ビット浮動小数点
System.Boolean	xmsBOOL	True/False (真/偽) の値
適用外	xmsCHAR	符号付きまたは符号なし 8 ビット値 (符号付きか符号なしかはプラットフォームによる)
System.Decimal	適用外	96 ビット符号付き整数 $\times 10^0$ から $10^{28}$ まで
System.Object	適用外	すべての型の基本
System.String	適用外	ストリング型

## ドット・ネットにおける管理操作および非管理操作

管理コードは、ドット・ネット 共通言語ランタイム環境の中で排他的に実行され、そのランタイムによって提供されるサービスに完全に依存します。アプリケーションが非管理に分類されるのは、そのアプリケーションの一部がドット・ネット 共通言語ランタイム環境の外部で実行されるか、または外部にあるサービスを呼び出す場合です。

拡張機能の中には、現在のところ管理ドット・ネット環境の中ではサポートできないものがあります。

完全管理環境で現在サポートされていない機能をアプリケーションで使用する必要がある場合は、そのアプリケーションに実質的な変更を加えることなく、そのアプリケーションが非管理環境を使用するように

変更することができます。ただし、その場合は XMS スタックで非管理コードが使用されることに注意してください。

## IBM WebSphere MQ キュー・マネージャー への接続

WMQ\_CM\_CLIENT への管理接続で、SSL 接続、非 TCP 通信、およびチャネル圧縮はサポートされていません。しかし、それらの接続は、非管理接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することによってサポートされる場合があります。詳しくは、[.NET アプリケーションの開発](#)を参照してください。

非管理環境で管理対象オブジェクトから接続ファクトリーを作成する場合、接続モードの値を手動で XMSC\_WMQ\_CM\_CLIENT\_UNMANAGED に変更する必要があります。

## WebSphere サービス統合バス メッセージング・エンジンとの接続

SSL プロトコル (HTTPS を含む) を使用することが必要な WebSphere サービス統合バス・メッセージング・エンジンとの接続は、現在のところ管理コードとしてサポートされていません。

### 関連資料

[XMSC WMQ CONNECTION\\_MODE](#)

## .NET での宛先

ドット・ネットでは、宛先はプロトコル・タイプに従って作成されるため、作成対象のプロトコル・タイプでのみ使用できます。

宛先を作成するための 2 つの関数が用意されており、1 つはトピック用、もう 1 つはキュー用です。

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

これらの関数は、API にある次の 2 つのオブジェクトで使用できます。

- `ISession`
- `XMSFactoryFactory`

どちらの場合も、これらのメソッドは、以下のフォーマットでパラメーターを含めることができる URI スタイル・ストリングを受け入れることができます。

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

あるいは、これらのメソッドは宛先名のみ (つまり、`topic://` または `queue://` のプレフィックスとパラメーターのない名前) を解釈することもできます。

したがって、URI スタイルのストリング

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

は、次の宛先名と同じ結果になります。

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

WebSphere サービス統合バス JMS に関しては、`topicname` と `topicspace` を両方指定するがパラメーターは指定できない簡略形式でトピックも指定できます。

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

## ドット・ネットでのプロパティ

ドット・ネット アプリケーションは、オブジェクトのプロパティを取得および設定するときに、`PropertyContext` インターフェースのメソッドを使用します。

[PropertyContext](#) インターフェースは、プロパティを取得して設定するメソッドをカプセル化します。これらのメソッドは、直接的または間接的に以下のクラスによって継承されます。

- [BytesMessage](#)
- [接続](#)
- [ConnectionFactory](#)
- [ConnectionMetaData](#)
- [Destination](#)
- [MapMessage](#)
- [Message](#)
- [MessageConsumer](#)
- [MessageProducer](#)
- [ObjectMessage](#)
- [QueueBrowser](#)
- [セッション](#)
- [StreamMessage](#)
- [TextMessage](#)

アプリケーションがプロパティの値を設定した場合、プロパティに設定されていた以前の値は新しい値によって置き換えられます。

XMS プロパティについて詳しくは、[181 ページ](#)の『[XMS オブジェクトのプロパティ](#)』を参照してください。

使い勝手を向上させるため、XMS での ドット・ネット プロパティの名前と値は、XMSC という構造体の public 定数として事前定義されます。これらの定数の名前は、XMSC.<constant>の形式です (例: XMSC.USERID (プロパティ名定数) および XMSC.DELIVERY\_AS\_APP (値定数))。

さらに、IBM.XMS.MQC 構造体を使用することによって IBM WebSphere MQ の定数にアクセスすることができます。IBM.XMS 名前空間は既にインポートされています。これらのプロパティの値には、MQC.<constant>の形式でアクセスできます。例えば、MQC.MQRO\_COA\_WITH\_FULL\_DATA

さらに、ドット・ネットの XMS ドット・ネット クラスと IBM WebSphere MQ クラスの両方を使用し、IBM.XMS および IBM.WMQ 名前空間の場合は、MQC 構造体の名前空間を完全に修飾して、各オカレンスが固有になるようにする必要があります。

現在のところ、高度な機能の一部は、管理 ドット・ネット 環境の中でサポートされていません。詳しくは、[47 ページ](#)の『[ドット・ネットにおける管理操作および非管理操作](#)』を参照してください。

## ドット・ネット での存在しないプロパティの処理

XMS ドット・ネット での存在しないプロパティの処理は、JMS 仕様と広範囲に整合しており、同時に XMS の C および C++ の実装環境とも整合しています。

JMS では、存在しないプロパティにアクセスした場合、存在しない (ヌル) 値を要求された型にメソッドが変換しようとする、Java システム例外が発生する可能性があります。プロパティが存在しない場合は、以下の例外が発生します。

- `getStringProperty` および `getObjectProperty` がヌルを戻す
- `Boolean.valueOf(null)` が `false` を戻すため、`getBooleanProperty` が `false` を戻す
- `Integer.valueOf(null)` が例外をスローするため、`getIntProperty` などが `java.lang.NumberFormatException` をスローする

プロパティが XMS ドット・ネット に存在しない場合は、以下の例外が発生します。

- `GetStringProperty` および `GetObjectProperty` (さらに `GetBytesProperty`) がヌルを戻す (Java の場合と同じ)
- `GetBooleanProperty` が `System.NullReferenceException` をスローする

- `GetIntProperty` などが `System.NullReferenceException` をスローする

この実装環境は Java とは異なりますが、JMS 仕様と広範囲に整合しており、同時に XMS C および C++ のインターフェースとも整合しています。Java 実装環境の場合と同様に、XMS ドット・ネットは、`System.Convert` 呼び出しから呼び出し元までのすべての例外に波及します。ただし、Java とは異なり、XMS は、ヌルをシステム変換ルーチンに渡すことでドット・ネットフレームワークの固有の動作を単に使用するのではなく、`NullReferenceExceptions` を明示的にスローします。アプリケーションがプロパティを "abc" のような String に設定し、`GetIntProperty` を呼び出すと、`Convert.ToInt32("abc")` によってスローされた `System.FormatException` は呼び出し元に波及します。この振る舞いは Java と一致しています。`MessageFormatException` がスローされるのは、`setProperty` と `getProperty` に使用した型が非互換である場合に限られます。この振る舞いも Java と一致しています。

## ドット・ネットでのエラー処理

XMS ドット・ネット例外は、すべて `System.Exception` から派生します。XMS メソッド呼び出しは、`MessageFormatException` などの特定の XMS 例外、一般的な `XMSEException`、または `NullReferenceException` などのシステム例外をスローすることができます。

特定の catch ブロックまたは汎用の `System.Exception` catch ブロックでこれらのエラーをキャッチするためのアプリケーションを作成します (どちらを使用するかは、アプリケーションの要件によって異なります)。

## ドット・ネットでのメッセージ・リスナーおよび例外リスナー

ドット・ネットアプリケーションは、メッセージ・リスナーを使用してメッセージを非同期に受信し、例外リスナーを使用して接続の問題に関する通知を非同期に受信します。

メッセージ・および例外リスナーの両方の機能は、ドット・ネットと C++ の場合と同じです。ただし、いくつかの実装に小さな相違点があります。

## ドット・ネットでのメッセージ・リスナー

メッセージを非同期に受信するには、以下の手順に従ってください。

1. メッセージ・リスナー代行のシグニチャーを突き合わせるメソッドを定義します。静的メソッドまたはインスタンス・メソッドのいずれかを定義できます。また、アクセス可能なクラスであればどのクラスでもメソッドを定義できます。代行シグニチャーは以下のとおりです。

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

また、メソッドを以下のように定義できます。

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

2. 次のようなコードを使用して、このメソッドを代行としてインスタンス化します。

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

3. 代行を 1 つ以上のコンシューマーに登録するため、コンシューマーの `MessageListener` プロパティに代行を次のように設定します。

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

`MessageListener` をヌルに戻すことで、代行を除去できます。

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

## .NET での例外リスナー

例外リスナーは、メッセージ・リスナーとほぼ同様に機能しますが、代行定義が異なり、メッセージ・コンシューマーではなく接続に割り当てられています。これは C++ と同一です。

1. メソッドを定義します。代行シグニチャーは以下のとおりです。

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

このため、定義されるメソッドは以下のようになります。

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

2. 以下のようなコードを使用して、このメソッドを代行としてインスタンス化します。

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

3. 代行を接続に登録するため、ExceptionListener プロパティを設定します。

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

ExceptionListener を次のようにリセットすることで、代行を除去できます。

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

例外またはメッセージは、それに対する参照が残っていなければ、システム・ガーベッジ・コレクターによって自動的に削除されます。

以下はサンプル・コードです。

```
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);
    }
}
```

## 管理対象オブジェクトでの作業

この節章では、管理対象オブジェクトについての情報を提供します。XMS アプリケーションは、中央の管理対象オブジェクト・リポジトリからオブジェクト定義を取得し、それらを使用して接続ファクトリーと宛先を作成できます。

この節章では、管理対象オブジェクトの作成と管理に役立つ情報を提供し、XMS によってサポートされる管理対象オブジェクト・リポジトリの種類について説明します。節章ではまた、XMS アプリケーションが管理対象オブジェクト・リポジトリへの接続を作成して、必要な管理対象オブジェクトを取得する方法についても説明します。

節章には、以下のトピック節が含まれています。

### 関連タスク

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

## サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

ファイル・システム・オブジェクト・ディレクトリーは、シリアライズされた Java and Naming Directory Interface (JNDI) オブジェクトの形式を使用します。LDAP オブジェクト・ディレクトリーは、JNDI オブジェクトを収容する任意のディレクトリーです。ファイル・システムと LDAP オブジェクト・ディレクトリーは、IBM WebSphere MQ v6.0 で提供されている JMSAdmin ツールまたは WebSphere MQ v7.0 以降で提供されている WebSphere MQ Explorer を使用して管理できます。ファイル・システムと LDAP オブジェクト・ディレクトリーの両方を使用して、IBM WebSphere MQ 接続ファクトリーと宛先を集中化することにより、クライアント接続を管理できます。ネットワーク管理者は、同じ中央リポジトリを参照し、中央リポジトリの接続設定に行われた変更を反映するために自動的に更新される、複数のアプリケーションをデプロイすることができます。

COS ネーミング・ディレクトリーは、WebSphere サービス統合バス 接続ファクトリーおよび宛先を含んでおり、WebSphere ApplicationServer 管理コンソールを使用して管理できます。XMS アプリケーションが COS ネーミング・ディレクトリーからオブジェクトを取り出すには、JNDI ルックアップ Web サービスがデプロイされている必要があります。この Web サービスは、すべての WebSphere サービス統合テクノロジーで使用できるわけではありません。詳しくは、製品資料を参照してください。

**注:** オブジェクト・ディレクトリーに対する変更を有効にするには、アプリケーション接続を再始動します。

### 関連概念

#### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

#### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

#### 管理対象オブジェクト

管理対象オブジェクトを使用すると、クライアント・アプリケーションが使用する接続設定を中央のリポジトリから管理できます。アプリケーションは、中央のリポジトリからオブジェクト定義を取り出して使用することにより、ConnectionFactory オブジェクトや Destination オブジェクトを作成できます。管理対象オブジェクトを使用すれば、アプリケーションと、アプリケーションが実行時に使用するリソースとを切り離すことができます。

#### 関連タスク

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

##### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

#### 関連資料

##### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

##### 管理対象 Destination オブジェクトの必須プロパティー

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティーを設定する必要があります。

## 管理対象オブジェクトのプロパティー・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティーを、定義に対応する XMS プロパティーで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティーにマップする必要があります。

例えば、IBM WebSphere MQ JMS 接続ファクトリーから取得したプロパティーを使用して XMS 接続ファクトリーを作成するには、この 2 つの間でプロパティーをマップする必要があります。

すべてのプロパティー・マッピングは、自動的に実行されます。

次の表には、接続ファクトリーおよび宛先の最も一般的なプロパティーのいくつかの間のマッピングを示します。この表に示すプロパティーはほんの数組の例に過ぎず、示されているすべてのプロパティーがすべての接続タイプおよびサーバーに関連するわけではありません。

IBM WebSphere MQ JMS プロパティー名	XMS プロパティー名
PERSISTENCE (PER)	XMSC_DELIVERY_MODE
EXPIRY (EXP)	XMSC_TIME_TO_LIVE
PRIORITY (PRI)	XMSC_PRIORITY

表 9. 接続ファクトリーと宛先のプロパティーの名前マッピングの例

IBM WebSphere MQ JMS プロパティー名	XMS プロパティー名	WebSphere サービス統合バス プロパティー名
PERSISTENCE (PER)	XMSC_DELIVERY_MODE	
EXPIRY (EXP)	XMSC_TIME_TO_LIVE	
PRIORITY (PRI)	XMSC_PRIORITY	
	XMSC_WPM_HOST_NAME	serverName
	XMSC_WPM_BUS_NAME	busName
	XMSC_WPM_TOPIC_SPACE	topicName

## 関連概念

### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### InitialContext プロパティー

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

## 関連タスク

### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## 関連資料

### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

### 管理対象 Destination オブジェクトの必須プロパティー

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティーを設定する必要があります。

### IDestination (.NET インターフェイス用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

### ICollectionFactory (.NET インターフェース用)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

### ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

以下の表にリストするプロパティは、メッセージング・サーバーへの接続を作成するためにアプリケーションで設定する必要がある最小限のプロパティです。接続の作成方法をカスタマイズする場合は、ご使用のアプリケーションで、必要に応じて ConnectionFactory オブジェクトの任意の追加プロパティを設定できます。詳しくは、183 ページの『ConnectionFactory のプロパティ』を参照してください。有効なプロパティの完全なリストが含まれています。

## IBM WebSphere MQ キュー・マネージャーへの接続

必要な XMS	必要となる同等の IBM WebSphere MQ JMS プロパティ
<u>XMSC_CONNECTION_TYPE</u>	XMS は、接続ファクトリーのクラス名と TRANSPORT (TRAN) プロパティからこのプロパティを解決します。
<u>XMSC_WMQ_HOST_NAME</u>	HOSTNAME (HOST)
<u>XMSC_WMQ_PORT</u>	PORT
<u>XMSC_WMQ_QUEUE_MANAGER</u>	キュー・マネージャーの名前

## ブローカーへのリアルタイム接続

必要な XMS	必要となる同等の IBM WebSphere MQ JMS プロパティ
<u>XMSC_CONNECTION_TYPE</u>	XMS は、接続ファクトリーのクラス名と TRANSPORT (TRAN) プロパティからこのプロパティを解決します。
<u>XMSC_RTT_HOST_NAME</u>	HOSTNAME (HOST)
<u>XMSC_RTT_PORT</u>	PORT

## WebSphere サービス統合バスへの接続

XMS プロパティ	説明
<u>XMSC_CONNECTION_TYPE</u>	アプリケーションの接続先となるメッセージング・サーバーのタイプです。これは、接続ファクトリーのクラス名から判別されます。

表 12. WebSphere サービス統合バスへの接続で使用する管理対象 *ConnectionFactory* オブジェクトのプロパティ設定 (続き)

XMS プロパティ	説明
XMSC.WPM.BUS.NAME	接続ファクトリーの場合は、アプリケーションの接続先となるサービス統合バスの名前であり、宛先の場合は、その宛先が存在するサービス統合バスの名前です。

## 関連概念

### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

### IBM WebSphere MQ キュー・マネージャーへのセキュア接続

XMS .NET アプリケーションが、IBM WebSphere MQ キュー・マネージャーとのセキュア接続を確立できるようにするには、関係するプロパティが ConnectionFactory オブジェクトで定義されていることが必要です。

### WebSphere サービス統合バス メッセージング・エンジンとのセキュア接続

XMS .NET アプリケーションが WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続を確立できるようにするには、関係するプロパティが ConnectionFactory オブジェクトで定義されていることが必要です。

## 関連タスク

### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## 関連資料

管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

#### IConnectionFactory (.NET インターフェース用)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

#### ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## 管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

接続のタイプ	プロパティ	説明
IBM WebSphere MQ キュー・マネージャー	QUEUE (QU)	接続先のキュー
	TOPIC (TOP)	アプリケーションが宛先として使用するトピック
ブローカーへのリアルタイム接続	TOPIC (TOP)	アプリケーションが宛先として使用するトピック

接続のタイプ	プロパティ	説明
IBM WebSphere MQ キュー・マネージャー	QUEUE (QU)	接続先のキュー
	TOPIC (TOP)	アプリケーションが宛先として使用するトピック
ブローカーへのリアルタイム接続	TOPIC (TOP)	アプリケーションが宛先として使用するトピック
WebSphere サービス統合バス	topicName	アプリケーションがトピックに接続している場合
	queueName	アプリケーションがキューに接続している場合

### 関連概念

#### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

#### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

#### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリーから管理対象オブジェクトを取り出します。

#### **関連タスク**

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

##### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリーへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

#### **関連資料**

##### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

##### IDestination (.NET インターフェース用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

##### Destination のプロパティー

Destination オブジェクトのプロパティーの概要と、詳細な参照情報へのリンクを示します。

## **管理対象オブジェクトの作成**

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### **始める前に**

XMS によってサポートされるさまざまな種類の管理対象オブジェクト・リポジトリーについて詳しくは、52 ページの『[サポートされる管理対象オブジェクト・リポジトリーのタイプ](#)』を参照してください。

### **このタスクについて**

IBM WebSphere MQ の管理対象オブジェクトを作成するには、IBM WebSphere MQ エクスプローラーまたは IBM WebSphere MQ JMS 管理 (JMSAdmin) ツールを使用します。

IBM WebSphere MQ、WebSphere Event Broker、または WebSphere Message Broker の管理対象オブジェクトを作成するには、IBM WebSphere MQ JMS 管理 (JMSAdmin) ツールを使用します。

WebSphere サービス統合バスの管理対象オブジェクトを作成するには、WebSphere ApplicationServer 管理コンソールを使用します。

以下のステップで、管理対象のオブジェクトを作成するために行うことを要約します。

### **手順**

1. 接続ファクトリーを作成し、必要なプロパティーを定義して、アプリケーションから選択対象のサーバーへの接続を作成します。

接続を作成するために XMS で必要とされる最低限のプロパティーは、55 ページの『[管理対象 ConnectionFactory オブジェクトの必須プロパティー](#)』で定義されています。

2. アプリケーションの接続先のメッセージング・サーバーで、必要な宛先を作成します。

- IBM WebSphere MQ キュー・マネージャーへの接続の場合は、キューまたはトピックを1つ作成します。
- ブローカーへのリアルタイム接続の場合は、トピックを1つ作成します。
- WebSphere サービス統合バスへの接続の場合は、キューまたはトピックを1つ作成します。

接続を作成するために XMS で必要とされる最低限のプロパティは、[57 ページ](#)の『[管理対象 Destination オブジェクトの必須プロパティ](#)』で定義されています。

## 関連概念

### [サポートされる管理対象オブジェクト・リポジトリのタイプ](#)

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### [管理対象オブジェクトのプロパティ・マッピング](#)

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

### [InitialContext プロパティ](#)

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

### [XMS 初期コンテキストの URI フォーマット](#)

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

### [JNDI ルックアップ Web サービス](#)

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

### [管理対象オブジェクトの検索](#)

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

### [管理対象オブジェクト](#)

管理対象オブジェクトを使用すると、クライアント・アプリケーションが使用する接続設定を中央のリポジトリから管理できます。アプリケーションは、中央のリポジトリからオブジェクト定義を取り出して使用することにより、ConnectionFactory オブジェクトや Destination オブジェクトを作成できます。管理対象オブジェクトを使用すれば、アプリケーションと、アプリケーションが実行時に使用するリソースとを切り離すことができます。

### [管理対象オブジェクトでの作業](#)

この節章では、管理対象オブジェクトについての情報を提供します。XMS アプリケーションは、中央の管理対象オブジェクト・リポジトリからオブジェクト定義を取得し、それらを使用して接続ファクトリーと宛先を作成できます。

### [ConnectionFactories オブジェクトと Connection オブジェクト](#)

ConnectionFactory オブジェクトには、アプリケーションが Connection オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、Connection オブジェクトを使用して Session オブジェクトを作成します。

### [WebSphere サービス統合バスへの接続](#)

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

## 関連タスク

### [InitialContext オブジェクト](#)

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## 関連資料

### 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

### 管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

### ConnectionFactory (.NET インターフェイス用)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

### ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

### IDestination (.NET インターフェイス用)

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### Destination のプロパティ

Destination オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## このタスクについて

InitialContext オブジェクトは、そのリポジトリへの接続をカプセル化します。XMS API には、以下のタスクを実行するためのメソッドが用意されています。

- InitialContext オブジェクトの作成
- 管理対象オブジェクト・リポジトリ内で、管理対象オブジェクトを検索する。

InitialContext オブジェクトの作成について詳しくは、ドット・ネットの [113 ページ](#)の『InitialContext』および [192 ページ](#)の『InitialContext のプロパティ』を参照してください。

## 関連概念

### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

#### **関連タスク**

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### **関連資料**

##### 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

##### 管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

##### InitialContext (.NET インターフェース用)

アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリから取得したオブジェクト定義によってオブジェクトを作成します。

##### InitialContext のプロパティ

InitialContext オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## **InitialContext プロパティ**

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

JNDI および ドット・ネットの XMS 実装では、追加情報をコンストラクターへの環境 Hashtable で指定します。

管理対象オブジェクト・リポジトリのロケーションは、XMSC\_IC\_URL プロパティで定義します。このプロパティは、通常は Create 呼び出しに渡されますが、検索の前に別のネーミング・ディレクトリーに接続するように変更できます。FileSystem コンテキストまたは LDAP コンテキストの場合、このプロパティはディレクトリーのアドレスを定義します。COS ネーミングの場合、これは、これらのプロパティを使用して JNDI ディレクトリーに接続する Web サービスのアドレスです。

以下のプロパティは、JNDI ディレクトリーに接続する目的で使用するように、未変更のまま Web サービスに渡されます。

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

#### **関連概念**

##### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

##### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

### **関連タスク**

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

### **関連資料**

#### 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

#### 管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

#### InitialContext (.NET インターフェース用)

アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリから取得したオブジェクト定義によってオブジェクトを作成します。

#### InitialContext のプロパティ

InitialContext オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

## **XMS 初期コンテキストの URI フォーマット**

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

## **FileSystem コンテキスト**

FileSystem コンテキストの場合は、ファイル・システム・ベースのディレクトリーのロケーションが URL によって指定されます。URL の構造は、RFC 1738 の「*Uniform Resource Locators (URL)*」に定義されています。URL にはプレフィックス `file://` があり、プレフィックスの後の構文は、XMS が動作するシステム上で開くことができるファイルの有効な定義になっています。

この構文はプラットフォーム固有の構文にすることができます。また、分離文字として「/」または「&#xa5;」を使用できます。「&#xa5;」を使用した場合、分離文字はそれぞれ追加の「&#xa5;」を使用してエスケープする必要があります。こうすることにより、.NET フレームワークで、分離文字がそれ以降の文字列のエスケープ文字として解釈されるのを防止できます。

以下の例に、この構文が示されています。

```
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
```

## LDAP コンテキスト

LDAP コンテキストの場合、URL の基本構造は RFC 2255 (LDAP URL の形式) によって定義され、大文字と小文字をを区別しない接頭部である `ldap://` が付けられます。

正確な構文を次に示します。

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

この構文は RFC で定義されているものと同様ですが、属性、スコープ、フィルター、拡張子のいずれにも対応していません。

この構文の例を以下に示します。

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

## WSS コンテキスト

WSS コンテキストの場合、URL は、`http://` のプレフィックスが付く、Web サービス・エンドポイントの形式になります。

あるいは、`cosnaming://` または `wsvc://` というプレフィックスを使用することもできます。

これら 2 つのプレフィックスは、HTTP を介してアクセスされる URL を持つ WSS コンテキストを使用しているという意味として解釈されます。WSS コンテキストを使用すると、URL から直接かつ容易に初期コンテキスト・タイプを得ることができます。

この構文例には、以下が含まれています。

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

## 関連概念

### サポートされる管理対象オブジェクト・リポジトリのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリーから管理対象オブジェクトを取り出します。

#### **関連タスク**

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

##### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリーへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

#### **関連資料**

##### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

##### 管理対象 Destination オブジェクトの必須プロパティー

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティーを設定する必要があります。

##### InitialContext (.NET インターフェース用)

アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリーから取得したオブジェクト定義によってオブジェクトを作成します。

##### InitialContext のプロパティー

InitialContext オブジェクトのプロパティーの概要と、詳細な参照情報へのリンクを示します。

## **JNDI ルックアップ Web サービス**

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

この Web サービスは、インストール・ディレクトリーにあるエンタープライズ・アーカイブ・ファイル SIBXJndiLookupEAR.ear に含まれています。Message Service Client for .NET の現行リリースの場合、SIBXJndiLookupEAR.ear は <install\_dir>\java\lib ディレクトリーにあります。管理コンソールまたは wsadmin スクリプト・ツールのいずれかを使用して、WebSphere サービス統合バス・サーバーにインストールできます。Web サービス・アプリケーションのデプロイについての詳細は、製品資料を参照してください。

XMS アプリケーションの内部で Web サービスを定義するために必要なことは、InitialContext オブジェクトの `XMSC_IC_URL` プロパティーを Web サービスのエンドポイント URL に設定することだけです。例えば、この Web サービスを MyServer というサーバー・ホストにデプロイする場合は、Web サービス・エンドポイント URL を以下のように設定します。

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

`XMSC_IC_URL` プロパティーを設定すると、InitialContext ルックアップの呼び出しがその定義済みのエンドポイントにある Web サービスを呼び出し、その Web サービスが COS ネーミング・サービスから必要な管理対象オブジェクトを検索するようになります。

ドット・ネット アプリケーションもこの Web サービスを使用できます。サーバー・サイドのデプロイメントは、XMS C、/C++ の場合も、XMS ドット・ネットの場合も同じです。XMS ドット・ネットは、Microsoft ドット・ネット フレームワークを介して Web サービスを直接呼び出します。

#### **関連概念**

サポートされる管理対象オブジェクト・リポジトリーのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

#### 管理対象オブジェクトのプロパティー・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティーを、定義に対応する XMS プロパティーで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティーにマップする必要があります。

#### InitialContext プロパティー

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリーのロケーションが含まれます。アプリケーションがリポジトリーへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリーのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリーから管理対象オブジェクトを取り出します。

### **関連タスク**

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリーへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

#### インストール・ウィザードを使用した Message Service Client for .NET のインストール

インストールには、InstallShield X/Windows MSI インストーラーが使用されます。完全インストールまたはカスタム・インストールのいずれかを選択できるように、2つのセットアップ・オプションが使用できます。

### **関連資料**

#### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

#### 管理対象 Destination オブジェクトの必須プロパティー

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティーを設定する必要があります。

## **管理対象オブジェクトの検索**

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティーに指定されているアドレスを使用して、リポジトリーから管理対象オブジェクトを取り出します。

検索するオブジェクトには、次のタイプの名前を指定できます。

- Destination オブジェクトを説明する単純名。例えば、SalesOrders というキューの宛先。
- 複合名。サブコンテキストで構成でき、「/」で区切られ、末尾はオブジェクト名にする必要があります。複合名の例は、「Warehouse/PickLists/DispatchQueue2」です。Warehouse および Picklists はネーミング・ディレクトリーのサブコンテキスト、DispatchQueue2 は Destination オブジェクトの名前を表します。

### **関連概念**

サポートされる管理対象オブジェクト・リポジトリーのタイプ

XMS は、ファイル・システム、Lightweight Directory Access Protocol (LDAP)、COS ネーミングの 3 種類の管理対象オブジェクト・ディレクトリーをサポートしています。管理対象オブジェクトの中でも、ファイル・システムと LDAP は IBM WebSphere MQ および WebSphere ApplicationServer への接続に使用できるのに対し、COS ネーミングは WebSphere ApplicationServer のみへの接続に使用できます。

#### 管理対象オブジェクトのプロパティー・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティーを、定義に対応する XMS プロパティーで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティーにマップする必要があります。

#### InitialContext プロパティー

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリーのロケーションが含まれます。アプリケーションがリポジトリーへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリーのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### JNDI ルックアップ Web サービス

XMS から COS ネーミング・ディレクトリーにアクセスするには、WebSphere サービス統合バス・サーバーに JNDI ルックアップ Web サービスをデプロイする必要があります。この Web サービスは、COS ネーミング・サービスから取得した Java 情報を、XMS アプリケーションが読み取れる形式に変換します。

### **関連タスク**

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリーへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

### **関連資料**

#### 管理対象 ConnectionFactory オブジェクトの必須プロパティー

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティーを定義する必要があります。

#### 管理対象 Destination オブジェクトの必須プロパティー

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティーを設定する必要があります。

#### InitialContext (.NET インターフェース用)

アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリーから取得したオブジェクト定義によってオブジェクトを作成します。

#### InitialContext のプロパティー

InitialContext オブジェクトのプロパティーの概要と、詳細な参照情報へのリンクを示します。

## **XMS アプリケーションの通信の保護**

この節章では、XMS アプリケーションが Secure Sockets Layer (SSL) により WebSphere サービス統合バスメッセージング・エンジンまたは IBM WebSphere MQ キュー・マネージャーに接続するためのセキュア通信をセットアップする方法について説明します。

この節章では、アプリケーションがセキュア接続を確立できるようにするための XMS ConnectionFactory プロパティーの構成方法について説明します。

節章には、以下のトピック節が含まれています。

## IBM WebSphere MQ キュー・マネージャー へのセキュア接続

XMS .NET アプリケーションが、IBM WebSphere MQ キュー・マネージャーとのセキュア接続を確立できるようにするには、関係するプロパティーが `ConnectionFactory` オブジェクトで定義されていることが必要です。

暗号化ネゴシエーションで使用されるプロトコルは、Secure Sockets Layer (SSL) または Transport Layer Security (TLS) のいずれかです。そのどちらを使用するかは、`ConnectionFactory` オブジェクトの中で指定されている `CipherSuite` によります。

IBM WebSphere MQ バージョン 7.0.0.1 およびそれ以降のクライアント・ライブラリーを使用して、IBM WebSphere MQ バージョン 7 のキュー・マネージャーに接続している場合は、XMS アプリケーションで同じキュー・マネージャーに対する複数の接続を作成できます。ただし、異なるキュー・マネージャーへの接続は許可されていません。接続しようとすると、`MQRC_SSL_ALREADY_INITIALIZED` エラーが発生します。

IBM WebSphere MQ バージョン 6 以降のクライアント・ライブラリーを使用している場合は、まず、それ以前の SSL 接続をすべてクローズした場合のみ、SSL 接続を作成できます。同じプロセスから同じキュー・マネージャーまたは異なるキュー・マネージャーに、複数の並行 SSL 接続を確立することは許可されていません。複数の要求をしようとすると、`MQRC_SSL_ALREADY_INITIALIZED` が出力されます。その場合、SSL 接続について要求されたパラメーターの一部が無視される可能性があります。

SSL を介した IBM WebSphere MQ マネージャーへの接続のための `ConnectionFactory` プロパティーを、簡単な説明とともに下の表に示します。

プロパティー名	説明
<code>XMSC_WMQ_SSL_CERT_STORES</code>	キュー・マネージャーとの SSL 接続で使用される証明書取り消しリスト (CRL) を保持するサーバーの位置。
<code>XMSC_WMQ_SSL_CIPHER_SPEC</code>	キュー・マネージャーとのセキュア接続で使用する <code>CipherSpec</code> の名前。
<code>XMSC_WMQ_SSL_CIPHER_SUITE</code>	キュー・マネージャーへの SSL 接続または TLS 接続で使用される <code>CipherSuite</code> の名前。セキュア接続のネゴシエーションで使用されるプロトコルは、指定されている <code>CipherSuite</code> によって異なります。
<code>XMSC_WMQ_SSL_CRYPTO_HW</code>	クライアント・システムに接続されている暗号ハードウェアに関する構成詳細情報。
<code>XMSC_WMQ_SSL_FIPS_REQUIRED</code>	このプロパティーの値は、アプリケーションが非 FIPS 準拠の暗号スイートを使用できるかどうかを決定します。このプロパティーが <code>true</code> (真) に設定されている場合、クライアント/サーバー接続には FIPS アルゴリズムだけが使用されます。
<code>XMSC_WMQ_SSL_KEY_REPOSITORY</code>	鍵および証明書が保存されている鍵データベース・ファイルの位置。
<code>XMSC_WMQ_SSL_KEY_RESETCOUNT</code>	<code>KeyResetCount</code> は、秘密鍵の再ネゴシエーションが実行されるまで、1 つの SSL 会話の中で送受信される暗号化されていないデータの合計バイト数を表します。
<code>XMSC_WMQ_SSL_PEER_NAME</code>	キュー・マネージャーとの SSL 接続で使用されるピア名。

### 関連資料

[IConnectionFactory \(.NET インターフェース用\)](#)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

[ConnectionFactory のプロパティー](#)

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

#### 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

### IBM WebSphere MQ キュー・マネージャー への接続のための CipherSuite および CipherSpec 名前マッピング

InitialContext は、JMSAdmin Connection Factory のプロパティ SSLCIPHERSUITE と、ほぼそれに相当する XMSC の XMSC\_WMQ\_SSL\_CIPHER\_SPEC との間の変換を実行します。

XMSC\_WMQ\_SSL\_CIPHER\_SUITE には値を指定したが、XMSC\_WMQ\_SSL\_CIPHER\_SPEC の値は省略した場合、類似の変換が必要になります。

68 ページの表 16 に、使用可能な CipherSpecs と、それらに相当する JSSE CipherSuite のリストを示します。

CipherSpec	相当する JSSE CipherSuite
DES_SHA_EXPORT	SSL_RSA_WITH_DES_CBC_SHA
DES_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
FIPS_WITH_3DES_EDE_CBC_SHA	SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
FIPS_WITH_DES_CBC_SHA	SSL_RSA_FIPS_WITH_DES_CBC_SHA
NULL_MD5	SSL_RSA_WITH_NULL_MD5
NULL_SHA	SSL_RSA_WITH_NULL_SHA
RC2_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
RC4_56_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
RC4_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC4_40_MD5
RC4_MD5_US	SSL_RSA_WITH_RC4_128_MD5
RC4_SHA_US	SSL_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	SSL_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA	SSL_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA	SSL_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA	SSL_RSA_WITH_DES_CBC_SHA
TRIPLE_DES_SHA_US	SSL_RSA_WITH_3DES_EDE_CBC_SHA

注: CipherSuite の名前 SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA または SSL\_RSA\_WITH\_DES\_CBC\_SHA の 1 対 1 のマッピングでは、プロパティ XMSC\_WMQ\_SSL\_FIPSREQUIRED の設定を考慮に入れ、ヒューリスティックを適用する必要があります。

プロパティ XMSC\_WMQ\_SSL\_CIPHER\_SUITE に SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA または SSL\_RSA\_WITH\_DES\_CBC\_SHA を指定し、XMSC\_WMQ\_SSL\_CIPHER\_SPEC に値がない場合、XMSC\_WMQ\_SSL\_CIPHER\_SPEC の値は、以下の表に従って選択されます。

XMSC\_WMQ\_SSL\_CIPHER\_SUITE プロパティに SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA を指定したときに XMSC\_WMQ\_SSL\_CIPHER\_SPEC に使用される値を、以下の表に示します。

表 17. XMSC_WMQ_SSL_CIPHER_SUITE プロパティに SSL_RSA_WITH_3DES_EDE_CBC_SHA を指定した場合に XMSC_WMQ_SSL_CIPHER_SPEC のために使用される値	
入力: XMSC_WMQ_SSL_FIPSREQUIRED 値	出力: 選択される XMSC_WMQ_SSL_CIPHER_SPEC
false (つまり MQSSL_FIPS_NO)	TRIPLE_DES_SHA_US
true (つまり MQSSL_FIPS_YES)	TLS_RSA_WITH_3DES_EDE_CBC_SHA

注:

- TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA は推奨されません。ただし、32 GB 以下のデータの転送にはまだ使用できますが、これを超えるとエラー AMQ9288 を出して接続が終了します。このエラーを回避するために、Triple-DES を使用しないか、またはこの CipherSpec を使用する際に秘密鍵リセットを有効にする必要があります。

XMSC\_WMQ\_SSL\_CIPHER\_SUITE プロパティに SSL\_RSA\_WITH\_DES\_CBC\_SHA を指定したときに XMSC\_WMQ\_SSL\_CIPHER\_SPEC に使用される値を、以下の表に示します。

表 18. XMSC_WMQ_SSL_CIPHER_SUITE プロパティに SSL_RSA_WITH_DES_CBC_SHA を指定した場合に XMSC_WMQ_SSL_CIPHER_SPEC のために使用される値	
入力: XMSC_WMQ_SSL_FIPSREQUIRED 値	出力: 選択される XMSC_WMQ_SSL_CIPHER_SPEC
false (つまり MQSSL_FIPS_NO)	DES_SHA_EXPORT
true (つまり MQSSL_FIPS_YES)	TLS_RSA_WITH_DES_CBC_SHA

## WebSphere サービス統合バス メッセージング・エンジンとのセキュア接続

XMS .NET アプリケーションが WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続を確立できるようにするには、関係するプロパティが ConnectionFactory オブジェクトで定義されていることが必要です。

XMS では、WebSphere サービス統合バスとの接続について SSL と HTTPS がサポートされています。SSL と HTTPS により、認証と機密性を考慮したセキュア接続が提供されます。

WebSphere セキュリティーの場合と同じように、XMS のセキュリティは、JSSE セキュリティー標準規格および命名規則に準拠して構成されています。それには、セキュア接続のネゴシエーション時に使用されるアルゴリズムを指定するための CipherSuite の使用も含まれています。暗号化ネゴシエーションで使用されるプロトコルは、SSL または TLS のいずれかです。そのどちらを使用するかは、ConnectionFactory オブジェクトの中で指定されている CipherSuite によります。

注: .NET アプリケーションの場合、セキュリティ機能は Microsoft Secure Channel (SChannel) によって提供されます。

69 ページの表 19 に、ConnectionFactory オブジェクトで定義する必要のあるプロパティを示します。

表 19. WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続に関する ConnectionFactory のプロパティ	
プロパティ名	説明
<b>XMSC_WPM_SSL_CIPHER_SUITE</b>	WebSphere サービス統合バス メッセージング・エンジンとの SSL または TLS 接続で使用される CipherSuite の名前。セキュア接続のネゴシエーションで使用されるプロトコルは、指定されている CipherSuite によって異なります。 注: このプロパティは .NET アプリケーションでサポートされています。

表 19. WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続に関する *ConnectionFactory* のプロパティ (続き)

プロパティ名	説明
<b>XMSC_WPM_SSL_KEYRING_LABEL</b>	サーバーによる認証で使用される証明書。 注: このプロパティは .NET アプリケーションでサポートされています。

以下に、WebSphere 統合メッセージング・エンジンとのセキュア接続に関する *ConnectionFactory* プロパティの例を示します。

```
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);
```

ここで *chain\_name* を、*BootstrapTunneledSecureMessaging* または *BootstrapSecureMessaging* のいずれかに設定する必要があります。さらに、*port\_number* は、ブートストラップ・サーバーが着信要求を listen するポートの番号です。

以下に、サンプル値が挿入された WebSphere 統合メッセージング・エンジンへのセキュア接続に関する *ConnectionFactory* のプロパティの例を示します。

```
/* 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");
```

## 関連資料

### [IConnectionFactory \(.NET インターフェース用\)](#)

アプリケーションは、接続ファクトリーを使用して接続を作成します。

### [ConnectionFactory のプロパティ](#)

*ConnectionFactory* オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

### [管理対象 ConnectionFactory オブジェクトの必須プロパティ](#)

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

## WebSphere サービス統合バスとの接続に関する *CipherSuite* および *CipherSpec* 名前マッピング

GSKit では *CipherSuite* ではなく *CipherSpec* が使用されるため、*XMSC\_WPM\_SSL\_CIPHER\_SUITE* プロパティに指定される JSSE スタイルの *CipherSuite* 名を GSKit スタイルの *CipherSpec* 名にマップする必要があります。

70 ページの表 20 に、認識された *CipherSuite* のそれぞれに対応する *CipherSpec* を示します。

CipherSuite	CipherSpec で対応するもの
SSL_RSA_WITH_NULL_MD5	NULL_MD5
SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5	RC2_MD5_EXPORT
SSL_RSA_EXPORT_WITH_RC4_40_MD5	RC4_MD5_EXPORT
SSL_RSA_WITH_RC4_128_MD5	RC4_MD5_US

CipherSuite	CipherSpec に対応するもの
SSL_RSA_WITH_NULL_SHA	NULL_SHA
SSL_RSA_EXPORT1024_WITH_RC4_56_SHA	RC4_56_SHA_EXPORT1024
SSL_RSA_WITH_RC4_128_SHA	RC4_SHA_US
SSL_RSA_WITH_DES_CBC_SHA	DES_SHA_EXPORT
SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA	DES_SHA_EXPORT1024
SSL_RSA_FIPS_WITH_DES_CBC_SHA	FIPS_WITH_DES_CBC_SHA
SSL_RSA_WITH_3DES_EDE_CBC_SHA	TRIPLE_DES_SHA_US
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA	FIPS_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA	TLS_RSA_WITH_DES_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA	TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA	TLS_RSA_WITH_AES_256_CBC_SHA

注:

- TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA は推奨されません。ただし、32 GB 以下のデータの転送にはまだ使用できますが、これを超えるとエラー AMQ9288 を出して接続が終了します。このエラーを回避するために、Triple-DES を使用しないか、またはこの CipherSpec を使用する際に秘密鍵リセットを有効にする必要があります。

## XMS メッセージ

この節章では、XMS メッセージの構造とコンテンツについて説明するとともに、アプリケーションによる XMS メッセージの処理方法について説明します。

この節章は、以下のトピック節から構成されます。

- [71 ページの『XMS メッセージのパーツ』](#)
- [72 ページの『XMS メッセージのヘッダー・フィールド』](#)
- [73 ページの『XMS メッセージのプロパティ』](#)
- [76 ページの『XMS メッセージの本文』](#)
- [83 ページの『メッセージ・セレクター』](#)
- [84 ページの『XMS メッセージの IBM WebSphere MQ メッセージへのマッピング』](#)

### 関連資料

[IMessage \(.NET インターフェイス用\)](#)

メッセージ・オブジェクトは、アプリケーションが送信または受信するメッセージを表します。IMessage は、IMapMessage などの Message クラスのスーパークラスです。

## XMS メッセージのパーツ

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

### ヘッダー

メッセージのヘッダーにはフィールドが含まれており、すべてのメッセージには同じセットのヘッダー・フィールドが含まれています。XMS とアプリケーションは、ヘッダー・フィールドの値を使用して経路メッセージを識別します。ヘッダー・フィールドについては、[72 ページの『XMS メッセージのヘッダー・フィールド』](#)を参照してください。

## プロパティのセット

メッセージのプロパティは、メッセージについての追加情報を指定します。すべてのメッセージには、同じセットのヘッダー・フィールドがありますが、すべてのメッセージはプロパティの異なるセットを持つことができます。詳しくは、[73 ページの『XMS メッセージのプロパティ』](#)を参照してください。

## Body

メッセージの本文にはアプリケーション・データが含まれています。詳しくは、[76 ページの『XMS メッセージの本文』](#)を参照してください。

アプリケーションは、受信するメッセージを選択できます。そのために、メッセージ・セレクターを使用して選択基準を指定します。選択基準は、特定のヘッダー・フィールドの値とメッセージの任意のプロパティの値に基づいて指定できます。メッセージ・セレクターについて詳しくは、[83 ページの『メッセージ・セレクター』](#)を参照してください。

## 関連資料

### [XMS メッセージのヘッダー・フィールド](#)

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

### [XMS メッセージのプロパティ](#)

XMS は、3 種類のメッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

### [XMS メッセージの本文](#)

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれずに、ヘッダー・フィールドとプロパティのみで構成されることもあります。

### [メッセージ・セレクター](#)

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

### [XMS メッセージの IBM WebSphere MQ メッセージへのマッピング](#)

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

## XMS メッセージのヘッダー・フィールド

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

これらのヘッダー・フィールドの名前は、プレフィックス JMS で始まります。JMS メッセージ・ヘッダー・フィールドについて詳しくは、「[Java Message Service Specification、バージョン 1.1](#)」を参照してください。

XMS は、JMS メッセージのヘッダー・フィールドを Message オブジェクトの属性としてインプリメントします。各ヘッダー・フィールドには、値の設定および取得に使用する独自のメソッドがあります。これらのメソッドについては、[125 ページの『IMessage』](#)を参照してください。ヘッダー・フィールドは、常時読み取り可能と書き込み可能です。

[72 ページの表 21](#) に、JMS メッセージのヘッダー・フィールドのリストと、伝送されるメッセージに設定される各フィールドの値を示します。一部のフィールドは、アプリケーションがメッセージを送信するとき、または JMSRedelivered の場合にはアプリケーションがメッセージを受信するときに、XMS によって自動的に設定されます。

JMS メッセージのヘッダー・フィールド名	送信されたメッセージ用の値の設定方法 (方法 [クラス] の形式)
JMSCorrelationID	JMSCorrelationID の設定 [Message]
JMSDeliveryMode	送信 [MessageProducer]
JMSDestination	送信 [MessageProducer]

表 21. JMS メッセージのヘッダー・フィールド (続き)

JMS メッセージのヘッダー・フィールド名	送信されたメッセージ用の値の設定方法 (方法 [クラス] の形式)
JMSExpiration	送信 [MessageProducer]
JMSMessageID	送信 [MessageProducer]
JMSPriority	送信 [MessageProducer]
JMSRedelivered	受信 [MessageConsumer]
JMSReplyTo	JMSReplyTo の設定 [Message]
JMSTimestamp	送信 [MessageProducer]
JMSType	JMSType の設定 [Message]

## 関連資料

### XMS メッセージのパーツ

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

### XMS メッセージのプロパティ

XMS は、3 種類のメッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

### XMS メッセージの本文

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれず、ヘッダー・フィールドとプロパティのみで構成されることもあります。

### メッセージ・セレクター

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

### XMS メッセージの IBM WebSphere MQ メッセージへのマッピング

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

## XMS メッセージのプロパティ

XMS は、3 種類のメッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

XMS アプリケーションは、WebSphere JMS アプリケーションとの間でメッセージを交換できます。これは、XMS が、次に示す Message オブジェクトの事前定義プロパティをサポートするためです。

- WebSphere JMS がサポートする JMS 定義プロパティと同じもの。このプロパティの名前は、プレフィックス JMSX で始まります。
- WebSphere JMS がサポートする IBM 定義プロパティと同じもの。このプロパティの名前は、プレフィックス JMS\_IBM\_ で始まります。

それぞれの事前定義プロパティには、2 つの名前があります。

- JMS 名 (JMS 定義プロパティの場合) または WebSphere JMS 名 (IBM 定義プロパティの場合)。

これは、JMS または WebSphere JMS でプロパティが認識される時の名前であり、このプロパティを持つメッセージとともに伝送される名前でもあります。XMS アプリケーションは、この名前を使用してメッセージ・セレクター式のプロパティを識別します。

- メッセージ・セレクター式を除くすべての状況でプロパティを識別するための XMS 名。各 XMS 名は、IBM.XMS.XMSC クラスで名前付き定数として定義されています。名前付き定数の値は、対応する JMS または WebSphere JMS 名になります。

事前定義プロパティに加えて、XMS アプリケーションは、メッセージ・プロパティの独自のセットを作成および使用できます。これらのプロパティは、アプリケーション定義プロパティと呼ばれます。

アプリケーションがメッセージを作成した後のメッセージのプロパティは、読み取り可能と書き込み可能です。プロパティは、アプリケーションがメッセージを送信した後も、読み取り可能と書き込み可能の状態のままになります。アプリケーションがメッセージを受信するときのメッセージのプロパティは、読み取り専用です。メッセージのプロパティが読み取り専用の場合にアプリケーションが `Message` クラスの `Clear Properties` メソッドを呼び出すと、プロパティは読み取り可能および書き込み可能になります。このメソッドにより、プロパティもクリアされます。

受信したメッセージを、メッセージ・プロパティのクリア後に転送するときの動作は、メッセージ・プロパティがクリアされた標準の `WMQ XMS for .NET ByteMessage` を転送するときの動作と整合しています。

ただし、次のプロパティが失われるため、これは推奨されません。

- `JMS_IBM_Encoding` プロパティ値。このプロパティがないと、意味を持つようにメッセージ・データをデコードできなくなります。
- `JMS_IBM_Format` プロパティ値。このプロパティがないと、(MQMD または新規の MQRFH2) メッセージ・ヘッダーと既存のヘッダーとの間のヘッダー・チェーニングが壊れます。

## 関連資料

### XMS メッセージのパーツ

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

### XMS メッセージのヘッダー・フィールド

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

### XMS メッセージの本文

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれずに、ヘッダー・フィールドとプロパティのみで構成されることもあります。

### メッセージ・セレクター

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

### XMS メッセージの IBM WebSphere MQ メッセージへのマッピング

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

## メッセージの JMS 定義プロパティ

XMS と WebSphere JMS の両方で、1つのメッセージに対し複数の JMS 定義プロパティがサポートされています。

74 ページの表 22 に、メッセージの JMS 定義プロパティのリストを示します。このリストにあるプロパティは、XMS と WebSphere JMS の両方でサポートされます。JMS 定義プロパティについては、*Java Message Service Specification*、バージョン 1.1 を参照してください。JMS 定義プロパティは、ブローカーへのリアルタイム接続では無効です。

この表では、各プロパティのデータ型を明示し、伝送されるメッセージに設定されるプロパティの値を示しています。一部のプロパティは、アプリケーションがメッセージを送信するとき、または `JMSXDeliveryCount` の場合にはアプリケーションがメッセージを受信するときに、XMS によって自動的に設定されます。

JMS 定義プロパティの XMS 名	JMS 名	データ・タイプ	伝送されるメッセージに設定される値 (形式は <code>method [class]</code> )
<code>JMSX_APPID</code>	<code>JMSXAppID</code>	<code>System.String</code>	送信 [ <code>MessageProducer</code> ]
<code>JMSX_DELIVERY_COUNT</code>	<code>JMSXDeliveryCount</code>	<code>System.Int32</code>	受信 [ <code>MessageConsumer</code> ]
<code>JMSX_GROUPID</code>	<code>JMSXGroupID</code>	<code>System.String</code>	ストリング・プロパティの設定 [ <code>PropertyContext</code> ]

表 22. メッセージの JMS 定義プロパティ (続き)

JMS 定義プロパティの XMS 名	JMS 名	データ・タイプ	伝送されるメッセージに設定される値 (形式は <i>method [class]</i> )
JMSX_GROUPSEQ	JMSXGroupSeq	System.Int32	整数の設定プロパティ [PropertyContext]
JMSX_USERID	JMSXUserID	System.String	送信 [MessageProducer]

### メッセージの IBM 定義プロパティ

XMS および WebSphere JMS では、1 つのメッセージに対して複数の IBM 定義プロパティがサポートされています。

75 ページの表 23 に、XMS と WebSphere JMS の両方によってサポートされる、メッセージの IBM 定義プロパティのリストを示します。IBM 定義プロパティについては、「*IBM WebSphere MQ Java の使用*」または WebSphere ApplicationServer 製品資料を参照してください。

この表では、各プロパティのデータ型を明示し、伝送されるメッセージに設定されるプロパティの値を示しています。一部のプロパティは、アプリケーションがメッセージを送信するときに、XMS によって自動的に設定されます。

表 23. メッセージの IBM 定義プロパティ

IBM 定義プロパティの XMS 名	WebSphere JMS 名	データ・タイプ	伝送されるメッセージに設定される値 (形式は <i>method [class]</i> )
JMS_IBM_CHARACTER_SET	JMS_IBM_Character_Set	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_ENCODING	JMS_IBM_Encoding	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_EXCEPTIONMESSAGE	JMS_IBM_ExceptionMessage	System.String	受信 [MessageConsumer]
JMS_IBM_EXCEPTIONREASON	JMS_IBM_ExceptionReason	System.Int32	受信 [MessageConsumer]
JMS_IBM_EXCEPTIONTIMESTAMP	JMS_IBM_ExceptionTimestamp	System.Int64	受信 [MessageConsumer]
JMS_IBM_EXCEPTIONPROBLEMDESTINATION	JMS_IBM_ExceptionProblemDestination	System.String	受信 [MessageConsumer]
JMS_IBM_FEEDBACK	JMS_IBM_Feedback	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_FORMAT	JMS_IBM_Format	System.String	ストリング・プロパティの設定 [PropertyContext]
JMS_IBM_LAST_MSG_IN_GROUP	JMS_IBM_Last_Msg_In_Group	System.Boolean	整数の設定プロパティ [PropertyContext]
JMS_IBM_MSGTYPE	JMS_IBM_MsgType	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_PUTAPPLTYPE	JMS_IBM_PutApplType	System.Int32	送信 [MessageProducer]
JMS_IBM_PUTDATE	JMS_IBM_PutDate	System.String	送信 [MessageProducer]

表 23. メッセージの IBM 定義プロパティ (続き)

IBM 定義プロパティの XMS 名	WebSphere JMS 名	データ・タイプ	伝送されるメッセージに設定される値 (形式は <i>method [class]</i> )
JMS_IBM_PUTTIME	JMS_IBM_PutTime	System.String	送信 [MessageProducer]
JMS_IBM_REPORT_COA	JMS_IBM_Report_COA	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_COD	JMS_IBM_Report_COD	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_DISCARD_MSG	JMS_IBM_Report_Discard_Msg	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_EXCEPTION	JMS_IBM_Report_Exception	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_EXPIRATION	JMS_IBM_Report_Expiration	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_NAN	JMS_IBM_Report_NAN	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_PAN	JMS_IBM_Report_PAN	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_PASS_CORREL_ID	JMS_IBM_Report_Pass_Correl_ID	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_REPORT_PASS_MSG_ID	JMS_IBM_Report_Pass_Msg_ID	System.Int32	整数の設定プロパティ [PropertyContext]
JMS_IBM_SYSTEM_MESSAGEID	JMS_IBM_System_MessageID	System.String	送信 [MessageProducer]

### メッセージのアプリケーション定義プロパティ

XMS アプリケーションは、メッセージ・プロパティの独自のセットを作成および使用できます。アプリケーションがメッセージを送信するときには、これらのプロパティもメッセージと一緒に伝送されます。受信側のアプリケーションは、メッセージ・セレクターを使用することにより、これらのプロパティの値に基づいて受信するメッセージを選択することができます。

WebSphere JMS アプリケーションが XMS アプリケーションから送信されたメッセージを選択して処理できるようにするには、「*IBM WebSphere MQ Java の使用*」で説明されているように、アプリケーション定義プロパティの名前が、メッセージ・セレクター式で ID を形成するための規則に準拠している必要があります。アプリケーション定義プロパティの値のデータ型は、System.Boolean、System.SByte、System.Int16、System.Int32、System.Int64、System.Float、System.Double、または System.String のいずれかである必要があります。

### XMS メッセージの本文

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれずに、ヘッダー・フィールドとプロパティのみで構成されることもあります。

XMS は、5 つのタイプのメッセージ本文をサポートします。

#### Bytes

本文にはバイト・ストリームが含まれています。この本文タイプのメッセージは、バイト・メッセージと呼ばれます。IBytesMessage インターフェースには、バイト・メッセージの本文を処理するメソッドが含まれています。詳しくは、78 ページの『[バイト・メッセージ](#)』を参照してください。

## マップ

本文には、名前値のペアが1組含まれており、それぞれの値には関連付けられたデータ型があります。この本文タイプのメッセージは、マップ・メッセージと呼ばれます。IMapMessage インターフェースには、マップ・メッセージの本文を処理するメソッドが含まれています。詳しくは、[79 ページの『マップ・メッセージ』](#)を参照してください。

## オブジェクト

本文には、シリアライズされた Java または ドット・ネット オブジェクトが含まれています。この本文タイプのメッセージは、オブジェクト・メッセージと呼ばれます。IObjectMessage インターフェースには、オブジェクト・メッセージの本文を処理するメソッドが含まれています。詳しくは、[80 ページの『オブジェクト・メッセージ』](#)を参照してください。

## ストリーム

本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。この本文タイプのメッセージは、ストリーム・メッセージと呼ばれます。IStreamMessage インターフェースには、ストリーム・メッセージの本文を処理するメソッドが含まれています。詳しくは、[81 ページの『ストリーム・メッセージ』](#)を参照してください。

## Text

本文にはストリングが含まれています。この本文タイプのメッセージは、テキスト・メッセージと呼ばれます。ITextMessage インターフェースには、テキスト・メッセージの本文を処理するメソッドが含まれています。詳しくは、[82 ページの『テキスト・メッセージ』](#)を参照してください。

IMessage インターフェースは、すべてのメッセージ・オブジェクトの親です。このインターフェースをメッセージング関数で使用することで、すべての XMS メッセージ・タイプを表すことができます。

それらの各データ型のサイズ、および最大値と最小値についての詳細は、[42 ページの表 5](#)を参照してください。

メッセージ本文に書き込まれるアプリケーション・データの要素に必要なデータ型について、また本文メッセージの5つのタイプについては、サブトピックを参照してください。

## 関連資料

### [XMS メッセージのパーツ](#)

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

### [XMS メッセージのヘッダー・フィールド](#)

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

### [XMS メッセージのプロパティ](#)

XMS は、3 種類のメッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

### [メッセージ・セレクター](#)

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

### [XMS メッセージの IBM WebSphere MQ メッセージへのマッピング](#)

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

## アプリケーション・データの要素のデータ型

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できる必要があります。

この理由により、XMS アプリケーションによってメッセージの本文に書き込まれるアプリケーション・データの各要素は、[78 ページの表 24](#) のリストにあるデータ型のいずれかを持っている必要があります。データ型ごとに、以下の表に互換性のある Java データ型を示します。XMS には、これらのデータ型のみを持つアプリケーション・データの要素を書き込むためのメソッドが用意されています。

表 24. Java データ型と互換性のある XMS データ型

XMS データ型	意味	互換性のある Java データ型
System.Boolean	ブール値 true または false	boolean
System.Char16	2 バイト文字	文字
System.SByte	符号付き 8 ビット整数	byte
System.Int16	符号付き 16 ビット整数	不足
System.Int32	符号付き 32 ビット整数	int
System.Int64	符号付き 64 ビット整数	long
System.Float	符号付き浮動小数点数	float
System.Double	符号付き倍精度浮動小数点数	double
System.String	文字ストリング	ストリング

これらの各データ型のサイズ、最大値、および最小値についての詳細は、42 ページの『XMS プリミティブ型』を参照してください。

### 関連概念

#### オブジェクトの属性とプロパティ

XMS オブジェクトには、オブジェクトの特性である属性とプロパティを設定できます。これらはさまざまな方法で実装されます。

#### XMS プリミティブ型

XMS には、Java の 8 個のプリミティブ型 (byte、short、int、long、float、double、char、および boolean) に相当するデータ型が用意されています。これにより、XMS と JMS の間で、データの損失や破損が発生することなくメッセージを交換することができます。

#### プロパティ値のデータ型の暗黙的な変換

アプリケーションがプロパティの値を取得するときに、XMS によりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMS がその変換を実行する方法を規定します。

### 関連資料

#### バイト・メッセージ

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

#### マップ・メッセージ

マップ・メッセージの本文には、名前値のペアが 1 組含まれており、それぞれの値には関連付けられたデータ型があります。

#### オブジェクト・メッセージ

オブジェクト・メッセージの本文には、シリアライズされた Java オブジェクトまたは .NET オブジェクトが含まれています。

#### ストリーム・メッセージ

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

#### テキスト・メッセージ

テキスト・メッセージの本文には、ストリングが含まれています。

### バイト・メッセージ

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

バイト・メッセージは、XMS アプリケーションが、XMS または JMS のアプリケーション・プログラミング・インターフェースを使用していないアプリケーションとメッセージを交換する必要がある場合に役立ちます。

アプリケーションがバイト・メッセージを作成した後のメッセージの本文は、書き込み専用です。アプリケーションは、ドット・ネットの `IBytesMessage` インターフェースの適切な書き込みメソッドを呼び出すことにより、アプリケーション・データを本文にアセンブルします。アプリケーションが値をバイト・メッセージ・ストリームに書き込むたびに、その値は、アプリケーションによって書き込まれた前の値の直後にアセンブルされます。XMS は、アセンブルされた最後のバイトの位置を記憶するために内部のカーソルを維持しています。

アプリケーションがメッセージを送信すると、メッセージの本文は読み取り専用になります。このモードのとき、アプリケーションはメッセージを繰り返し送信できます。

アプリケーションがバイト・メッセージを受信するときのメッセージの本文は、読み取り専用です。アプリケーションは、`IBytesMessage` インターフェースの適切な読み取りメソッドを使用して、バイト・メッセージ・ストリームのコンテンツを読み取ることができます。アプリケーションはバイトを順番に読み取り、XMS は、読み取られた最後のバイトの位置を記憶するために内部のカーソルを維持しています。

バイト・メッセージの本文が書き込み可能のときに、アプリケーションが `IBytesMessage` インターフェースの `Reset` メソッドを呼び出すと、その本文は読み取り専用になります。このメソッドはまた、バイト・メッセージ・ストリームの先頭でカーソルを位置変更します。

バイト・メッセージの本文が読み取り専用の場合に、アプリケーションがドット・ネットの `IMessage` インターフェースの `Clear Body` メソッドを呼び出すと、その本文は書き込み可能になります。このメソッドにより、本文もクリアされます。

## 関連資料

### アプリケーション・データの要素のデータ型

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

### マップ・メッセージ

マップ・メッセージの本文には、名前値のペアが 1 含まれており、それぞれの値には関連付けられたデータ型があります。

### オブジェクト・メッセージ

オブジェクト・メッセージの本文には、シリアライズされた Java オブジェクトまたは .NET オブジェクトが含まれています。

### ストリーム・メッセージ

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

### テキスト・メッセージ

テキスト・メッセージの本文には、ストリングが含まれています。

### IBytesMessage (.NET インターフェース用)

バイト・メッセージとは、本体がバイトのストリームからなるメッセージです。

## マップ・メッセージ

マップ・メッセージの本文には、名前値のペアが 1 含まれており、それぞれの値には関連付けられたデータ型があります。

それぞれの名前と値のペアにおいて、名前は値を識別するストリングであり、値は [78 ページの表 24](#) にリストされている XMS データ・タイプのいずれかを持つアプリケーション・データの要素です。名前値ペアの順序は定義されていません。MapMessage クラスには、名前値ペアを設定および取得するメソッドが含まれています。

アプリケーションは、その名前を指定することによって名前値ペアにランダムにアクセスできます。

ドット・ネット アプリケーションは、MapNames プロパティを使用して、マップ・メッセージの本文にある名前の列挙を取得することができます。

アプリケーションが名前と値のペアから値を取得するとき、その値は XMS によって別のデータ型に変換される可能性があります。例えば、マップ・メッセージの本文から整数を取得するには、アプリケーションは `MapMessage` クラスの `GetString` メソッドを呼び出すことで、その整数を文字列として取得することができます。サポートされる型変換は、XMS によってプロパティの値が 1 つのデータ型から別のデータ型に変換される場合にサポートされる型変換と同じです。サポートされる変換について詳しくは、[42 ページの『プロパティ値のデータ型の暗黙的な変換』](#)を参照してください。

アプリケーションがマップ・メッセージを作成した後のメッセージの本文は、読み取り可能と書き込み可能です。本文は、アプリケーションがメッセージを送信した後も、読み取り可能と書き込み可能な状態のままになります。アプリケーションがマップ・メッセージを受信するときのメッセージの本文は、読み取り専用です。マップ・メッセージの本文が読み取り専用のときに、アプリケーションが `Message` クラスの `Clear Body` メソッドを呼び出すと、本文は読み取り可能と書き込み可能になります。このメソッドにより、本文もクリアされます。

## 関連概念

### [プロパティ値のデータ型の暗黙的な変換](#)

アプリケーションがプロパティの値を取得するとき、XMS によりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMS がその変換を実行する方法を規定します。

## 関連資料

### [アプリケーション・データの要素のデータ型](#)

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

### [バイト・メッセージ](#)

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

### [オブジェクト・メッセージ](#)

オブジェクト・メッセージの本文には、シリアライズされた Java オブジェクトまたは .NET オブジェクトが含まれています。

### [ストリーム・メッセージ](#)

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

### [テキスト・メッセージ](#)

テキスト・メッセージの本文には、文字列が含まれています。

### [IMapMessage \(.NET インターフェース用\)](#)

マップ・メッセージとは、本体が名前と値のペアで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。

## オブジェクト・メッセージ

オブジェクト・メッセージの本文には、シリアライズされた Java オブジェクトまたは .NET オブジェクトが含まれています。

XMS アプリケーションは、オブジェクト・メッセージを受信してヘッダー・フィールドとプロパティを変更してから、そのメッセージを別の宛先に送信することができます。またアプリケーションは、オブジェクト・メッセージの本文をコピーし、そのコピーを使用して別のオブジェクト・メッセージを形成することもできます。XMS は、オブジェクト・メッセージの本文をバイトの配列として扱います。

アプリケーションがオブジェクト・メッセージを作成した後のメッセージの本文は、読み取り可能と書き込み可能です。本文は、アプリケーションがメッセージを送信した後も、読み取り可能と書き込み可能な状態のままになります。アプリケーションがオブジェクト・メッセージを受信するときのメッセージの本文は、読み取り専用です。オブジェクト・メッセージの本文が読み取り専用のときに、アプリケーションがドット・ネットの `IMessage` インターフェースの `Clear Body` メソッドを呼び出すと、その本文は読み取り可能および書き込み可能になります。このメソッドにより、本文もクリアされます。

## 関連資料

### [アプリケーション・データの要素のデータ型](#)

XMS アプリケーションが IBM WebSphere MQ classes for JMS アプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

#### バイト・メッセージ

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

#### マップ・メッセージ

マップ・メッセージの本文には、名前値のペアが 1 組含まれており、それぞれの値には関連付けられたデータ型があります。

#### ストリーム・メッセージ

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

#### テキスト・メッセージ

テキスト・メッセージの本文には、ストリングが含まれています。

#### IOBJECTMESSAGE (.NET インターフェース用)

オブジェクト・メッセージとは、本体が、直列化された Java または .NET オブジェクトからなるメッセージです。

### **ストリーム・メッセージ**

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

値のデータ型は、[78 ページの表 24](#) のリストにある XMS データ型のいずれかです。

アプリケーションがストリーム・メッセージを作成した後のメッセージの本文は、書き込み可能です。アプリケーションは、ドット・ネットの `IStreamMessage` インターフェースの適切な書き込みメソッドを呼び出すことにより、アプリケーション・データを本文にアセンブルします。アプリケーションが値をメッセージ・ストリームに書き込むたびに、その値とデータ型は、アプリケーションによって書き込まれた前の値の直後にアセンブルされます。XMS は、アセンブルされた最後の値の位置を記憶するために内部のカーソルを維持しています。

アプリケーションがメッセージを送信すると、メッセージの本文は読み取り専用になります。このモードのとき、アプリケーションはメッセージの送信を複数回実行することができます。

アプリケーションがストリーム・メッセージを受信するときのメッセージの本文は、読み取り専用です。アプリケーションは、ドット・ネットの `IStreamMessage` インターフェースの適切な読み取りメソッドを使用して、メッセージ・ストリームのコンテンツを読み取ることができます。アプリケーションは、値を順番に読み取り、XMS は、読み取られた最後の値の位置を記憶するために内部のカーソルを維持しています。

アプリケーションがメッセージ・ストリームから値を読み取る場合、その値は XMS によって別のデータ・タイプに変換されることがあります。例えば、メッセージ・ストリームから整数を読み取るには、アプリケーションは `ReadString` メソッドを呼び出すことで、その整数をストリングとして取得することができます。サポートされる型変換は、XMS によってプロパティの値が 1 つのデータ型から別のデータ型に変換される場合にサポートされる型変換と同じです。サポートされる変換について詳しくは、[42 ページの『プロパティ値のデータ型の暗黙的な変換』](#) を参照してください。

アプリケーションがメッセージ・ストリームから値を読み込もうとするときにエラーが発生すると、カーソルは次に進みません。アプリケーションは、別のデータ型として値の読み取りを試みることにより、エラーから回復できます。

ストリーム・メッセージの本文が書き込み専用のときに、アプリケーションがドット・ネットの `IStreamMessage` インターフェースの `Reset` メソッドを呼び出した場合、その本文は読み取り専用になります。このメソッドはまた、メッセージ・ストリームの先頭でカーソルを位置変更します。

ストリーム・メッセージの本文が読み取り専用の場合に、アプリケーションがドット・ネットの `IMessage` インターフェースの `Clear Body` メソッドを呼び出すと、その本文は書き込み専用になります。このメソッドにより、本文もクリアされます。

## 関連概念

[プロパティ値のデータ型の暗黙的な変換](#)

アプリケーションがプロパティの値を取得するときに、XMSによりこの値のデータ型を別のデータ型に変換できます。多くのルールが、サポートされる変換と、XMSがその変換を実行する方法を規定します。

## 関連資料

[アプリケーション・データの要素のデータ型](#)

XMSアプリケーションがIBM WebSphere MQ classes for JMSアプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

[バイト・メッセージ](#)

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

[マップ・メッセージ](#)

マップ・メッセージの本文には、名前値のペアが1組含まれており、それぞれの値には関連付けられたデータ型があります。

[オブジェクト・メッセージ](#)

オブジェクト・メッセージの本文には、シリアライズされたJavaオブジェクトまたは.NETオブジェクトが含まれています。

[テキスト・メッセージ](#)

テキスト・メッセージの本文には、ストリングが含まれています。

[IStreamMessage \(.NET インターフェース用\)](#)

ストリーム・メッセージとは、本体が値のストリームで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。本体の内容は、順番に読み書きされます。

## テキスト・メッセージ

テキスト・メッセージの本文には、ストリングが含まれています。

アプリケーションがテキスト・メッセージを作成した後のメッセージの本文は、読み取り可能と書き込み可能です。本文は、アプリケーションがメッセージを送信した後も、読み取り可能と書き込み可能な状態のままになります。アプリケーションがテキスト・メッセージを受信するときのメッセージの本文は、読み取り専用です。テキスト・メッセージの本文が読み取り専用のときに、アプリケーションがドット・ネットのIMessage インターフェースのClear Bodyメソッドを呼び出した場合、その本文は読み取り可能および書き込み可能になります。このメソッドにより、本文もクリアされます。

## 関連資料

[アプリケーション・データの要素のデータ型](#)

XMSアプリケーションがIBM WebSphere MQ classes for JMSアプリケーションとメッセージを交換できるようにするには、両方のアプリケーションが、メッセージの本文にあるアプリケーション・データを同じように解釈できることが必要です。

[バイト・メッセージ](#)

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

[マップ・メッセージ](#)

マップ・メッセージの本文には、名前値のペアが1組含まれており、それぞれの値には関連付けられたデータ型があります。

[オブジェクト・メッセージ](#)

オブジェクト・メッセージの本文には、シリアライズされたJavaオブジェクトまたは.NETオブジェクトが含まれています。

[ストリーム・メッセージ](#)

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

[ITextMessage \(.NET インターフェース用\)](#)

テキスト・メッセージとは、本体がストリングからなるメッセージです。

## メッセージ・セレクター

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

アプリケーションは、メッセージ・コンシューマーを作成する場合、メッセージ・セレクター式をそのコンシューマーに関連付けることができます。メッセージ・セレクター式により、選択基準を指定します。

アプリケーションが IBM WebSphere MQ V7.0 キュー・マネージャーに接続されている場合は、メッセージの選択はキュー・マネージャー側で行われます。XMS では選択を行わず、単にキュー・マネージャーから受信したメッセージを配信します。そのため、パフォーマンスが向上します。

ただし、アプリケーションの接続先が IBM WebSphere MQ V6.0 以下、WebSphere イベント・ブローカー、WebSphere メッセージ・ブローカー のいずれかになっている場合は、WebSphere サービス統合バス XMS によって、それぞれの着信メッセージが選択基準を満たしているかどうかを確認されます。メッセージが選択基準を満たす場合、XMS はメッセージをメッセージ・コンシューマーに配信します。メッセージが選択基準を満たさない場合、XMS はメッセージを配信せず、メッセージは Point-to-Point ドメインでキューに残されます。

アプリケーションは、複数のメッセージ・コンシューマーを作成し、それぞれに独自のメッセージ・セレクター式を設定することができます。着信メッセージが複数のメッセージ・コンシューマーの選択基準を満たす場合、XMS はそれらの各コンシューマーに着信メッセージを配信します。

メッセージ・セレクター式は、メッセージの次のプロパティを参照できます。

- JMS 定義プロパティ
- IBM 定義プロパティ
- アプリケーション定義プロパティ

次のメッセージ・ヘッダーのフィールドも参照できます。

- JMSCorrelationID
- JMSDeliveryMode
- JMSMessageID
- JMSPriority
- JMSTimestamp
- JMSType

ただし、メッセージ・セレクター式は、メッセージの本文のデータを参照できません。

次に、メッセージ・セレクター式の例を示します。

```
JMSPriority > 3 AND manufacturer = 'Jaguar' AND model in ('xj6','xj12')
```

XMS は、メッセージの優先順位が 3 より大きい場合にのみ、このメッセージ・セレクター式を使用してメッセージをメッセージ・コンシューマーに配信します。これは、値が `Jaguar;` のアプリケーション定義プロパティの製造元と、値が `xj6` または `xj12` の別のアプリケーション定義プロパティのモデルです。

XMS でメッセージ・セレクター式を形成する場合の構文規則は、IBM WebSphere MQ classes for JMS の場合と同じです。メッセージ・セレクター式の構成方法については、「*WebSphere MQ Java の使用*」を参照してください。メッセージ・セレクター式では、JMS 定義プロパティの名前が JMS 名になるように、また IBM 定義プロパティの名前が IBM WebSphere MQ classes for JMS 名になるようにしてください。メッセージ・セレクター式で、XMS 名を使用することはできません。

### 関連資料

#### [XMS メッセージのパーツ](#)

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

#### [XMS メッセージのヘッダー・フィールド](#)

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

#### [XMS メッセージのプロパティ](#)

XMS は、3 種類のメッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

#### XMS メッセージの本文

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれずに、ヘッダー・フィールドとプロパティのみで構成されることもあります。

#### XMS メッセージの IBM WebSphere MQ メッセージへのマッピング

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

## XMS メッセージの IBM WebSphere MQ メッセージへのマッピング

XMS メッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージのヘッダー構造のフィールドにマッピングされます。

XMS アプリケーションが IBM WebSphere MQ キュー・マネージャーに接続されている場合、キュー・マネージャーに送信されるメッセージは、IBM WebSphere MQ メッセージにマッピングされます。その方法は、類似の環境において IBM WebSphere MQ classes for JMS メッセージが IBM WebSphere MQ メッセージにマッピングされる場合と同じです。

#### Destination オブジェクトの XMSC\_WMQ\_TARGET\_CLIENT プロパティが

XMSC\_WMQ\_TARGET\_DEST\_JMS に設定されている場合、宛先に送信されるメッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージの MQMD および MQRFH2 ヘッダー構造のフィールドにマッピングされます。この方法で XMSC\_WMQ\_TARGET\_CLIENT プロパティを設定する場合は、メッセージを受信するアプリケーションが MQRFH2 ヘッダーを処理できることが前提となります。従って受信側のアプリケーションは、MQRFH2 ヘッダーを処理するように設計されている別の XMS アプリケーション、IBM WebSphere MQ classes for JMS アプリケーション、またはネイティブ IBM WebSphere MQ アプリケーションとなります。

#### Destination オブジェクトの XMSC\_WMQ\_TARGET\_CLIENT プロパティが代わりに

XMSC\_WMQ\_TARGET\_DEST\_MQ に設定されている場合、宛先に送信されるメッセージの JMS ヘッダー・フィールドとプロパティは、IBM WebSphere MQ メッセージの MQMD ヘッダー構造のフィールドにマッピングされます。メッセージには MQRFH2 ヘッダーが含まれておらず、MQMD ヘッダー構造のフィールドにマッピングできない JMS ヘッダー・フィールドとプロパティはすべて無視されます。したがって、そのメッセージを受信するアプリケーションは、MQRFH2 ヘッダーを処理するには設計されていないネイティブ IBM WebSphere MQ アプリケーションにすることができます。

キュー・マネージャーから受信される IBM WebSphere MQ メッセージは、XMS メッセージにマッピングされます。その方法は、類似の環境において IBM WebSphere MQ メッセージが IBM WebSphere MQ classes for JMS メッセージにマッピングされる場合と同じです。

着信 IBM WebSphere MQ メッセージに MQRFH2 ヘッダーがある場合、結果の XMS メッセージの本体のタイプは、MQRFH2 ヘッダーの mcd フォルダーに含まれる **Msd** プロパティの値によって決まります。

MQRFH2 ヘッダーに **Msd** プロパティが含まれていない場合、または IBM WebSphere MQ メッセージに MQRFH2 ヘッダーがない場合、出力される XMS メッセージには、MQMD ヘッダーの *Format* フィールドの値によって判別されるタイプの本文が含まれています。*Format* フィールドが MQFMT\_STRING に設定されていれば、XMS メッセージはテキスト・メッセージです。それ以外の場合、XMS メッセージはバイト・メッセージです。IBM WebSphere MQ メッセージに MQRFH2 ヘッダーがない場合は、MQMD ヘッダーのフィールドから派生可能な JMS ヘッダー・フィールドとプロパティのみが設定されます。

IBM WebSphere MQ classes for JMS メッセージから IBM WebSphere MQ メッセージへのマッピングについて詳しくは、*IBM WebSphere MQ Java* を参照してください。

### 関連資料

#### XMS メッセージのパーツ

XMS メッセージは、ヘッダー、プロパティのセット、および本文で構成されています。

#### XMS メッセージのヘッダー・フィールド

XMS アプリケーションが、WebSphere JMS アプリケーションとの間でメッセージを交換できるようにするため、XMS メッセージのヘッダーには JMS メッセージのヘッダー・フィールドが含まれています。

#### XMS メッセージのプロパティ

XMS は、3 種類の メッセージ・プロパティ (JMS 定義プロパティ、IBM 定義プロパティ、およびアプリケーション定義プロパティ) をサポートします。

#### XMS メッセージの本文

メッセージの本文にはアプリケーション・データが含まれています。ただしメッセージは、本文が含まれずに、ヘッダー・フィールドとプロパティのみで構成されることもあります。

#### メッセージ・セレクター

XMS アプリケーションは、メッセージ・セレクターを使用して、受信するメッセージを選択します。

### **Message Service Client for .NET アプリケーションからのメッセージ記述子の読み取りと書き込み**

IBM WebSphere MQ メッセージの StrucId と Version 以外のすべてのメッセージ記述子 (MQMD) フィールドにアクセスできます。BackoutCount は読み取り可能ですが、書き込むことはできません。この機能は、IBM WebSphere MQ キュー・マネージャーバージョン 6 以降に接続されている場合のみ使用可能であり、後で説明する宛先プロパティによって制御されます。

Message Service Client for .NET に用意されているメッセージ属性は、XMS アプリケーションが MQMD フィールドを設定したり、IBM WebSphere MQ アプリケーションを駆動したりするときに役立ちます。

publish/subscribe・メッセージングを使用する場合は、いくつかの制約事項が適用されます。例えば、MsgID や CorrelId などの MQMD フィールドは、設定しても無視されます。

このトピックで説明されている機能は、IBM WebSphere MQ V6 キュー・マネージャーに接続している場合、publish/subscribe メッセージングには使用できません。PROVIDERVERSION プロパティを 6 に設定している場合も使用できません。

### **Message Service Client for .NET アプリケーションからの IBM WebSphere MQ メッセージ・データへのアクセス**

MQRFH2 ヘッダー (存在する場合) および Message Service Client for .NET アプリケーション内のその他の IBM WebSphere MQ ヘッダー (存在する場合) を含む完全な IBM WebSphere MQ メッセージ・データに、JMSBytesMessage の本体としてアクセスできます。

このトピックで説明されている機能は、バージョン 7 以降の IBM WebSphere MQ キュー・マネージャーに接続されていて、かつ IBM WebSphere MQ メッセージング・プロバイダーが通常モードになっている場合のみ使用できます。

Destination オブジェクト・プロパティによって、XMS アプリケーションが JMSBytesMessage の本文として IBM WebSphere MQ メッセージ全体 (MQRFH2 ヘッダーが存在する場合は、このヘッダーも含む) にアクセスする方法が決定されます。

## **トラブルシューティング**

この節章では、Message Service Client for .NET の使用時における問題の検出とその対処に役立つ情報を提供します。

この節章では、XMS アプリケーションの問題判別に役立つ情報を提供し、ドット・ネットアプリケーションの First Failure Data Capture (FFDC) およびトレースを構成する方法について説明します。

この節章は、以下のトピック節から構成されます。

- [85 ページの『ドット・ネット アプリケーションのトレース構成』](#)
- [90 ページの『ドット・ネット アプリケーションの FFDC 構成』](#)
- [90 ページの『トラブルシューティングのヒント』](#)

## **ドット・ネット アプリケーションのトレース構成**

XMS ドット・ネット アプリケーションのトレースを構成するには、アプリケーション構成ファイルを使用するか、または XMS 環境変数を使用します。トレースするコンポーネントを選択できます。一般に、トレースは IBM サポートの指導に従って使用します。

XMS ドット・ネット のトレースは、標準 ドット・ネット トレース・インフラストラクチャーに基づいています。

デフォルトでは、エラー・トレースを除くすべてのトレースが使用不可に設定されています。以下のいずれかの方法で、トレースをオンにしてトレース設定を構成できます。

- アプリケーション構成ファイルを使用する。このファイルの名前は、ファイルに関連する実行可能プログラムの名前とサフィックス `.config` からなります。例えば、`text.exe` のアプリケーション構成ファイルの名前は `text.exe.config` になります。XMS ドット・ネット アプリケーションのトレースを使用可能にする方法としては、アプリケーション構成ファイルを使用する方法が推奨されます。詳しくは、[87 ページの『アプリケーション構成ファイルを使用したトレース構成』](#)を参照してください。
- XMS 環境変数を使用する (XMS C または C++ アプリケーションの場合)。詳しくは、[88 ページの『XMS 環境変数を使用したトレース構成』](#)を参照してください。

アクティブ・トレース・ファイルの名前の形式は `xms_trace <PID> .log` です。ここで、`<PID>` はアプリケーションのプロセス ID を表します。アクティブ・トレース・ファイルのサイズは、デフォルトでは 20 MB に制限されています。この制限に達すると、ファイルの名前が変更され、アーカイブされます。アーカイブ・ファイルの名前の形式は `xms_trace <PID> _YY.MM.DD_HH.MM.SS.log`

デフォルトでは、保存されるトレース・ファイルの数は 4 つ、つまり 1 つのアクティブ・ファイルと 3 つのアーカイブ・ファイルが保存されます。これらの 4 つのファイルは、アプリケーションが停止するまで循環バッファとして使用されます。この場合、最も古いファイルが削除され、最新ファイルに置き換えられます。トレース・ファイルの数を変更するには、アプリケーション構成ファイルで別の数を指定します。ただし、少なくとも 2 つのファイル (1 つのアクティブ・ファイルと 1 つのアーカイブ・ファイル) が必要です。

次の 2 種類のフォーマットのトレース・ファイルが使用可能です。

- `basic` フォーマット・トレース・ファイルは WebSphere Application Server フォーマットで人間が読み取り可能なファイルです。このフォーマットは、デフォルトのトレース・ファイル・フォーマットです。`basic` フォーマットには、トレース・アナライザー・ツールとの互換性がありません。
- `advanced` フォーマット・トレース・ファイルには、トレース・アナライザー・ツールとの互換性があります。`advanced` フォーマットのトレース・ファイルを作成する旨をアプリケーション構成ファイルで指定する必要があります。

トレース項目には、以下の情報が含まれています。

- トレースが記録された日時
- クラス名
- トレース・タイプ
- トレース・メッセージ

トレースの例を以下に示します。

```
[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
```

上記の例のフォーマットは以下のとおりです。

[Date Time:Microsecs] or Exit	Thread-id	Classname	Trace-type	Methodname	Entry
----------------------------------	-----------	-----------	------------	------------	-------

Trace-type は以下のとおりです。

- > (Entry)
- < (Exit)
- d (デバッグ情報)

## アプリケーション構成ファイルを使用したトレース構成

XMS ドット・ネット アプリケーションのトレース構成方法としては、アプリケーション構成ファイルを使用した構成が推奨されます。このファイルの trace セクションには、トレース対象を定義するパラメーター、トレース・ファイル位置と許容最大サイズ、使用されるトレース・ファイルの数、およびトレース・ファイルのフォーマットが記述されます。

アプリケーション構成ファイルを使用したトレースをオンにするために必要なのは、このファイルをアプリケーションの実行可能ファイルと同じディレクトリーに配置することだけです。

トレースはコンポーネント別およびトレース・タイプ別に使用可能にできます。トレース・グループ全体のトレースをオンにすることも可能です。階層内のコンポーネントごとにトレースをオンにすることも、その階層のすべてのコンポーネントのトレースをオンにすることもできます。使用可能なトレースのタイプを以下に示します。

- デバッグ・トレース
- 例外トレース
- 警告、通知メッセージ、およびエラー・メッセージ
- メソッドの入り口/出口トレース

アプリケーション構成ファイルの Trace セクションで定義されたトレース設定の例を以下に示します。

```
<?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>
```

87 ページの表 25 で、パラメーター設定について詳しく説明します。

パラメーター	説明
traceSpecification=<ComponentName>=<type>=<state>	<p>&lt;ComponentName&gt; は、トレースするクラスの名前です。この名前には、ワイルドカード文字 * を使用できます。例えば *=all=enabled はすべてのクラスをトレースすることを指定し、IBM.XMS.impl.*=all=enabled は API トレースのみが必要であることを指定します。</p> <p>&lt;type&gt; には以下のいずれかのトレース・タイプを指定できます。</p> <ul style="list-style-type: none"><li>• all</li><li>• debug</li><li>• イベント</li><li>• EntryExit</li></ul> <p>&lt;state&gt; は enabled または disabled のいずれかです。</p> <p>複数のトレース・エレメントを続けて入力するには、「:」（コロン）区切り文字を使用します。</p>

パラメーター	説明
traceFilePath="<filename>"	<p>traceFilePath を指定しないか、または traceFilePath が指定されているが空の文字列が含まれている場合には、トレース・ファイルは現行ディレクトリーに保管されます。トレース・ファイルを指定のディレクトリーに保管するには、traceFilePath にディレクトリー名を指定します。以下に例を示します。</p> <pre>traceFilePath="c:\somepath"</pre>
traceFileSize="<size>"	<p>トレース・ファイルの最大許容サイズ。ファイルは、このサイズに達すると、アーカイブされて名前変更されます。デフォルトの最大サイズは 20 KB です。これは、traceFileSize="20000000" と指定されます。</p>
traceFileNumber="<number>"	<p>保存対象のトレース・ファイルの数。デフォルトは 4 です (1 つのアクティブ・ファイルと 3 つのアーカイブ・ファイル)。最小数は 2 です。</p>
traceFormat="<format>"	<p>デフォルトのトレース・フォーマットは basic です。traceFormat="basic" を指定するか、traceFormat を指定しないか、または traceFormat を指定するが空の文字列が含まれている場合は、トレース・ファイルはこのフォーマットで作成されます。</p> <p>トレース・アナライザー・ツールと互換性のあるトレースが必要な場合は、traceFormat="advanced" を指定してください。</p>

アプリケーション構成ファイルのトレース設定は動的であり、ファイルが保管または置換されるたびに再読み取りされます。編集されたファイルでエラーが検出されると、トレース・ファイルの設定はデフォルト値に戻されます。

### 関連概念

#### XMS 環境変数を使用したトレース構成

アプリケーション構成ファイルを使用する代わりに、XMS 環境変数を使用してトレースをオンにすることもできます。これらの環境変数は、アプリケーション構成ファイルにトレース指定がない場合にのみ使用できます。

### XMS 環境変数を使用したトレース構成

アプリケーション構成ファイルを使用する代わりに、XMS 環境変数を使用してトレースをオンにすることもできます。これらの環境変数は、アプリケーション構成ファイルにトレース指定がない場合にのみ使用できます。

XMS ドット・ネット アプリケーションのトレースを構成するには、アプリケーションを実行する前に以下の環境変数を設定します。

環境変数	デフォルト	設定	意味
XMS_TRACE_ON	適用外	適用外: この変数の値は無視されます。	XMS_TRACE_ON が設定されている場合、デフォルトではすべてのトレースが使用可能になります。

表 26. .NET トレースの環境変数設定 (続き)

環境変数	デフォルト	設定	意味
XMS_TRACE_FILE_PATH	Current®作業ディレクトリー	/dirpath/	<p>トレースと FFDC レコードの書き込み先ディレクトリーのパス。</p> <p>XMS は、FFDC ファイルとトレース・ファイルを現行作業ディレクトリーに作成します (ただし、代わりの場所を指定する場合は別です)。代わりの場所を指定するには、XMS が FFDC ファイルとトレース・ファイルを作成するディレクトリーの完全修飾パス名を環境変数 XMS_TRACE_FILE_PATH に設定します。トレース対象のアプリケーションを開始する前に、この環境変数を設定しておく必要があります。アプリケーションを実行するユーザー ID に、XMS が FFDC ファイルとトレース・ファイルを作成するディレクトリーへの書き込み権限があることも確認しておかなければなりません。</p>
XMS_TRACE_FORMAT	BASIC	BASIC、ADVANCED	<p>必須トレース・フォーマット (BASIC または ADVANCED) を指定します。デフォルト・フォーマットは BASIC です。ADVANCED フォーマットには、トレース・アナライザー・ツールとの互換性があります。</p>
XMS_TRACE_SPECIFICATION	適用外	<p><a href="#">87 ページの『アプリケーション構成ファイルを使用したトレース構成』</a>を参照してください。</p>	<p><a href="#">87 ページの『アプリケーション構成ファイルを使用したトレース構成』</a>で指定されているフォーマットに従うトレース指定をオーバーライドします。</p>

### 関連概念

#### アプリケーション構成ファイルを使用したトレース構成

XMS ドット・ネット アプリケーションのトレース構成方法としては、アプリケーション構成ファイルを使用した構成が推奨されます。このファイルの trace セクションには、トレース対象を定義するパラメーター、トレース・ファイル位置と許容最大サイズ、使用されるトレース・ファイルの数、およびトレース・ファイルのフォーマットが記述されます。

## ドット・ネット アプリケーションの FFDC 構成

XMS のドット・ネット 実装では、FFDC ごとに 1 つの FFDC ファイルが作成されます。

First Failure Data Capture (FFDC) ファイルは、人が判読できるテキスト・ファイルの形で保管されています。これらのファイルの名前は、`xmsffdc<processID>_<Date>T<Timestamp>.txt` という形式になっています。ファイル名は、例えば `xmsffdc264_2006.01.06T13.18.52.990955.txt` のようになります。タイム・スタンプにはマイクロ秒の値まで含まれています。

ファイルの先頭には例外の発生日時が記述されており、その後には例外のタイプが続きます。これらのファイルには、固有の短い `probeId` も含まれています。この `probeId` に基づいて、この FFDC の発生位置を確認できます。

FFDC をオンにするための構成は必要ありません。デフォルトでは、すべての FFDC ファイルが現行ディレクトリーに書き込まれます。ただし、必要に応じて別のディレクトリーを指定することもできます。そのためには、アプリケーション構成ファイルの Trace セクションの `ffdcDirectory` を変更します。次に示す例では、すべてのトレース・ファイルがディレクトリー `c:\client\ffdc` に書き込まれます。

```
<IBM.XMS>
  <Trace ffdc=true ffdcDirectory="c:\client\ffdc"/>
</IBM.XMS>
```

トレースを使用不可にするには、アプリケーション構成ファイルの Trace セクションの FFDC を `false` に設定します。

アプリケーション構成ファイルを使用していない場合は、FFDC がオン、トレースがオフになっています。

## トラブルシューティングのヒント

次に挙げるヒントは、XMS を使用した問題のトラブルシューティングに役立ちます。

### XMS アプリケーションがキュー・マネージャーに接続できない (MQRC\_NOT\_AUTHORIZED)

XMS ドット・ネット クライアントの動作は、IBM WebSphere MQ JMS クライアントの動作とは異なる可能性があります。したがって、JMS アプリケーションはキュー・マネージャーに接続できるにもかかわらず、XMS アプリケーションがキュー・マネージャーに接続できないことがあります。

- この問題の簡単な解決方法は、長さが 12 文字以下のユーザー ID を使用し、キュー・マネージャーの権限リストの中で完全な許可を得ることです。この解決方法が理想的ではない場合、もっと複雑な別のアプローチとしてセキュリティ出口を使う方法が考えられます。この問題についてさらに詳しい情報が必要であれば、IBM サポートにご連絡ください。
- 接続ファクトリーの `XMSC_USERID` プロパティを設定している場合、それはログオン・ユーザーのユーザー ID およびパスワードと一致していなければなりません。このプロパティを設定しない場合、デフォルトのログオン・ユーザーのユーザー ID がキュー・マネージャーによって使用されます。
- IBM WebSphere MQ のユーザー認証は、`XMSC.USERID` フィールドと `XMSC.PASSWORD` フィールドにある情報ではなく、現在ログオンしているユーザーの詳細を使用して行われます。この目的は、IBM WebSphere MQ との整合性を維持することにあります。認証について詳しくは、オンラインの IBM WebSphere MQ 製品資料で「認証情報」を参照してください。

### 接続がメッセージング・エンジンにリダイレクトされる

WebSphere ApplicationServer バージョン 6.0.2 サービス統合バスに接続すると、元のプロバイダー・エンドポイントから、そのクライアント接続に対してバスが選択したメッセージング・エンジンに、すべての接続がリダイレクトされる場合があります。その場合は常に、IP アドレスによってではなくホスト名によって指定されたホスト・サーバーに接続がリダイレクトされます。したがって、ホスト名を解決できない場合は接続の問題が発生することがあります。

WebSphere ApplicationServer バージョン 6.0.2 サービス統合バスに正常に接続するためには、ホスト名と IP アドレスのマッピングをクライアント・ホスト・マシンに用意しなければならない場合があります。例えば、クライアント・ホスト・マシン上のローカル・ホスト・テーブルでマッピングを指定できます。

## XMS アプリケーションが使用する JVM ヒープが大きい

WebSphere ApplicationServer メッセージング・エンジンを介してメッセージを送信する XMS .NET アプリケーションは通常、指定されたデフォルトよりも大きい JVM ヒープを使用する必要があります。ヒープ構成設定を変更するには、WebSphere Application Server バージョン 7 製品資料のサービス統合テクノロジーを使用するメッセージングのパフォーマンスの調整を参照してください。

## Telnet のようなパスワード認証のサポート

XMS .NET Real Time Transport プロトコルは、Telnet のような単純なパスワード認証のみをサポートしません。XMS .NET Real Time Transport プロトコルは、保護品質 (QoP) をサポートしていません。

## double 型プロパティの値の設定

Windows 64 ビット・プラットフォームでは、double 型プロパティの値を設定または取得するとき、値が Double.Epsilon よりも小さい場合は、SetDoubleProperty() メソッドまたは GetDoubleProperty() メソッドが正しく機能しないことがあります。

例えば、double 型のプロパティに値 4.9E-324 を設定しようとしても、Windows 64 ビット・プラットフォームはそれを 0.0 として扱います。したがって、分散メッセージング環境では、JMS または別のアプリケーションが Unix または Windows の 32 ビット・マシン上で double 型プロパティの値を 4.9E-324 として設定した場合、XMS .Net が 64 ビット・マシン上で実行されていれば、GetDoubleProperty() から返される値は 0.0 になります。これは、Microsoft .NET 2.0 Framework の既知の問題です。

## Message Service Client for .NET のリファレンス

このリファレンス・セクションでは、Message Service Client for .NET を使用するとき役立つ情報を取り上げます。この情報は、XMS でのプログラミングに関連した作業に役立ちます。

## ドット・ネットインターフェース

このトピック節では、ドット・ネット クラスのインターフェースとそのプロパティおよびメソッドについて説明します。

以下の表に、すべてのインターフェースの要約を示します。これらのインターフェースは、IBM.XMS ネーム・スペースの範囲内で定義されます。

インターフェース	説明
<a href="#">94 ページの『IBytesMessage』</a>	バイト・メッセージとは、本体がバイトのストリームからなるメッセージです。
<a href="#">104 ページの『IConnection』</a>	Connection オブジェクトは、アプリケーションからメッセージング・サーバーへのアクティブな接続を表します。
<a href="#">107 ページの『IConnectionFactory』</a>	アプリケーションは、接続ファクトリーを使用して接続を作成します。
<a href="#">109 ページの『IConnectionMetaData』</a>	ConnectionMetaData オブジェクトは、接続に関する情報を提供します。
<a href="#">110 ページの『IDestination』</a>	宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。
<a href="#">112 ページの『ExceptionListener』</a>	アプリケーションは例外リスナーを使用して、接続の問題に関する通知を非同期に受信します。

表 27. ドット・ネット クラス・インターフェースの要約 (続き)

インターフェース	説明
<a href="#">112 ページの『IllegalStateException』</a>	XMS がこの例外をスローするのは、アプリケーションがメソッドを呼び出した時刻が誤っているか不適切である場合、あるいは要求した操作に対して XMS が適切な状態でない場合です。
<a href="#">113 ページの『InitialContext』</a>	アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリから取得したオブジェクト定義によってオブジェクトを作成します。
<a href="#">115 ページの『InvalidClientIDException』</a>	XMS がこの例外をスローするのは、アプリケーションが接続のクライアント ID を設定しようとしたが、そのクライアント ID が無効かまたは既使用中である場合です。
<a href="#">116 ページの『InvalidDestinationException』</a>	XMS がこの例外をスローするのは、アプリケーションが無効な宛先を指定している場合です。
<a href="#">116 ページの『InvalidSelectorException』</a>	XMS がこの例外をスローするのは、無効な構文のメッセージ・セレクター式をアプリケーションで指定した場合です。
<a href="#">116 ページの『IMapMessage』</a>	マップ・メッセージとは、本体が名前と値のペアで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。
<a href="#">125 ページの『IMessage』</a>	メッセージ・オブジェクトは、アプリケーションが送信または受信するメッセージを表します。IMessage は、IMapMessage などの Message クラスのスーパークラスです。
<a href="#">132 ページの『IMessageConsumer』</a>	アプリケーションは、メッセージ・コンシューマーを使用して、宛先に送信されたメッセージを受信します。
<a href="#">135 ページの『MessageEOFException』</a>	XMS がこの例外をスローするのは、アプリケーションがバイト・メッセージの本体を読み取っているときに、XMS がバイト・メッセージ・ストリームの終端を検出した場合です。
<a href="#">135 ページの『MessageFormatException』</a>	XMS がこの例外をスローするのは、XMS が無効なフォーマットのメッセージを検出した場合です。
<a href="#">135 ページの『IMessageListener (代行)』</a>	アプリケーションは、メッセージ・リスナーを使用して、メッセージを非同期に受信します。
<a href="#">136 ページの『MessageNotReadableException』</a>	XMS がこの例外をスローするのは、書き込み専用になっているメッセージの本体をアプリケーションが読み取ろうとした場合です。
<a href="#">136 ページの『MessageNotWritableException』</a>	XMS がこの例外をスローするのは、読み取り専用になっているメッセージの本体にアプリケーションが書き込もうとした場合です。
<a href="#">136 ページの『IMessageProducer』</a>	アプリケーションは、メッセージ・プロデューサーを使用して、メッセージを宛先に送信します。
<a href="#">142 ページの『IOBJECTMessage』</a>	オブジェクト・メッセージとは、本体が、直列化された Java または .NET オブジェクトからなるメッセージです。

表 27. ドット・ネット クラス・インターフェースの要約 (続き)

インターフェース	説明
143 ページの『 <a href="#">IPropertyContext</a> 』	IPropertyContext は、プロパティを取得および設定するメソッドを含む抽象スーパークラスです。これらのメソッドは、その他のクラスによって継承されます。
153 ページの『 <a href="#">IQueueBrowser</a> 』	アプリケーションは、キュー・ブラウザーを使用して、キュー上のメッセージを参照します。その際にメッセージは除去されません。
155 ページの『 <a href="#">要求者</a> 』	アプリケーションはリクエスターを使用して、要求メッセージを送信し、応答を待機して受信します。
156 ページの『 <a href="#">ResourceAllocationException</a> 』	XMS がこの例外をスローするのは、メソッドが必要とするリソースを XMS が割り振ることができない場合です。
156 ページの『 <a href="#">SecurityException</a> 』	XMS がこの例外をスローするのは、アプリケーションを認証するために指定したユーザー ID とパスワードがリジェクトされた場合です。XMS は、権限検査が不合格になり、そのためにメソッドを完了できない場合にもこの例外をスローします。
157 ページの『 <a href="#">ISession</a> 』	セッションとは、メッセージ送受信のための単一スレッドのコンテキストのことです。
168 ページの『 <a href="#">IStreamMessage</a> 』	ストリーム・メッセージとは、本体が値のストリームで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。
177 ページの『 <a href="#">ITextMessage</a> 』	テキスト・メッセージとは、本体がストリングからなるメッセージです。
178 ページの『 <a href="#">TransactionInProgressException</a> 』	XMS がこの例外をスローするのは、トランザクションが進行中であるために無効になっている操作をアプリケーションが要求した場合です。
178 ページの『 <a href="#">TransactionRolledBackException</a> 』	XMS がこの例外をスローするのは、アプリケーションが現行のトランザクションをコミットするために <code>Session.commit()</code> を呼び出したにもかかわらず、その後このトランザクションがロールバックされた場合です。
XMSC	ドット・ネットの場合、XMS プロパティの名前および値は、パブリック定数としてこのクラスに定義されます。詳しくは、 <a href="#">181 ページの『XMS オブジェクトのプロパティ』</a> を参照してください。
178 ページの『 <a href="#">XMSException</a> 』	<p>XMS がドット・ネット メソッドの呼び出しを処理しているときにエラーを検出すると、XMS は例外をスローします。例外とは、エラーに関する情報をカプセル化するオブジェクトのことです。</p> <p>XMS 例外にはさまざまなタイプがあり、XMSException オブジェクトは例外の 1 タイプに過ぎません。ただし、XMSException クラスは、その他の XMS 例外クラスのスーパークラスです。XMS は、XMSException オブジェクト以外のタイプの例外では適切でない状態では、XMSException オブジェクトをスローします。</p>

表 27. ドット・ネット クラス・インターフェースの要約 (続き)

インターフェース	説明
<a href="#">179 ページの『XMSFactoryFactory』</a>	アプリケーションが管理対象オブジェクトを使用していない場合は、このクラスを使用して接続ファクトリー、キュー、およびトピックを作成します。

各メソッドの定義では、XMS がメソッドの呼び出しの処理中にエラーを検出した場合に戻す例外コードをリストしています。各例外コードは、その名前付き定数で表されますが、この定数には対応する例外があります。

### 関連概念

[ユーザー独自のアプリケーションの作成](#)

ユーザー独自のアプリケーションをビルドする方法は、サンプル・アプリケーションをビルドする場合と同様です。

[XMS アプリケーションの書き込み](#)

この節章では、XMS アプリケーションを作成する場合に役立つ情報を提供します。

[XMS .NET アプリケーションの作成](#)

この節章では、XMS ドット・ネット アプリケーションを作成する場合に役立つ情報を提供します。

### 関連資料

[XMS オブジェクトのプロパティ](#)

この節章では、XMS で定義したオブジェクトのプロパティについて説明します。

## IByteMessage

バイト・メッセージとは、本体がバイトのストリームからなるメッセージです。

継承の階層:

```

IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.IByteMessage
    
```

### 関連資料

[バイト・メッセージ](#)

バイト・メッセージの本文には、バイトのストリームが含まれています。本文には実際のデータのみが含まれており、このデータを解釈する役割は、送受信を行うアプリケーションに委ねられています。

## ドット・ネット プロパティ

ドット・ネット プロパティの要約:

**ドット・ネット プロパティ 説明**  
**イ**

**BodyLength**                      メッセージの本体が読み取り専用である場合に、メッセージの本体の長さ (バイト単位) を取得します。

*BodyLength* - 本体の長さの取得

インターフェース:

```

Int64 BodyLength
{
    get;
}
    
```

メッセージの本体が読み取り専用である場合に、メッセージの本体の長さ (バイト単位) を取得します。

この値は、メッセージを読み取るためのカーソルの現在の位置にかかわらず、本体の全体の長さが戻されます。

#### 例外:

- XMSEException
- MessageNotReadableException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">ReadBoolean</a>	バイト・メッセージ・ストリームからブール値を読み取ります。
<a href="#">ReadSignedByte</a>	バイト・メッセージ・ストリームから、次のバイトを符号付き 8 ビット整数として読み取ります。
<a href="#">ReadBytes</a>	バイト・メッセージ・ストリームから、カーソルの現在位置から始まるバイトの配列を読み取ります。
<a href="#">ReadChar</a>	バイト・メッセージ・ストリームから、次の 2 バイトを文字として読み取ります。
<a href="#">ReadDouble</a>	バイト・メッセージ・ストリームから、次の 8 バイトを倍精度浮動小数点数として読み取ります。
<a href="#">ReadFloat</a>	バイト・メッセージ・ストリームから、次の 4 バイトを浮動小数点数として読み取ります。
<a href="#">ReadInt</a>	バイト・メッセージ・ストリームから、次の 4 バイトを符号付き 32 ビット整数として読み取ります。
<a href="#">ReadLong</a>	バイト・メッセージ・ストリームから、次の 8 バイトを符号付き 64 ビット整数として読み取ります。
<a href="#">ReadShort</a>	バイト・メッセージ・ストリームから、次の 2 バイトを符号付き 16 ビット整数として読み取ります。
<a href="#">ReadByte</a>	バイト・メッセージ・ストリームから、次のバイトを符号なし 8 ビット整数として読み取ります。
<a href="#">ReadUnsignedShort</a>	バイト・メッセージ・ストリームから、次の 2 バイトを符号なし 16 ビット整数として読み取ります。
<a href="#">ReadUTF</a>	バイト・メッセージ・ストリームから、UTF-8 でエンコードされたストリングを読み取ります。
<a href="#">Reset</a>	メッセージの本体を読み取り専用モードにして、カーソルをバイト・メッセージ・ストリームの先頭に位置変更します。
<a href="#">WriteBoolean</a>	バイト・メッセージ・ストリームへブール値を書き込みます。
<a href="#">WriteByte</a>	バイト・メッセージ・ストリームへ 1 バイト書き込みます。
<a href="#">WriteBytes</a>	バイト・メッセージ・ストリームへバイトの配列を書き込みます。
<a href="#">WriteBytes</a>	部分的なバイト配列を、指定の長さで定義したとおりにバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteChar</a>	文字を、上位バイトを先にして、2 バイトでバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteDouble</a>	倍精度浮動小数点数を長整数に変換し、その長整数を、上位バイトを先にして、8 バイトでバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteFloat</a>	浮動小数点数を整数に変換し、その整数を、上位バイトを先にして、4 バイトでバイト・メッセージ・ストリームに書き込みます。

メソッド	説明
<a href="#">WriteInt</a>	整数を、上位バイトを先にして、4バイトでバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteLong</a>	長整数を、上位バイトを先にして、8バイトでバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteObject</a>	指定したオブジェクトをバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteShort</a>	短整数を、上位バイトを先にして、2バイトでバイト・メッセージ・ストリームに書き込みます。
<a href="#">WriteUTF</a>	バイト・メッセージ・ストリームへ、UTF-8でエンコードされたストリングを書き込みます。

*ReadBoolean* - ブール値の読み取り

**インターフェース:**

```
Boolean ReadBoolean();
```

バイト・メッセージ・ストリームからブール値を読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られるブール値。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadSignedByte* - バイトの読み取り

**インターフェース:**

```
Int16 ReadSignedByte();
```

バイト・メッセージ・ストリームから、次のバイトを符号付き8ビット整数として読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られるバイト。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadBytes* - バイトの読み取り

**インターフェース:**

```
Int32 ReadBytes(Byte[] array);
Int32 ReadBytes(Byte[] array, Int32 length);
```

バイト・メッセージ・ストリームから、カーソルの現在位置から始まるバイトの配列を読み取ります。

## パラメーター:

### array (出力)

読み取られるバイトの配列を含むバッファー。呼び出しの前の、ストリームから読み取られる残りバイト数が、バッファーの長さより大きい場合、バッファーはいっぱいになります。残りバイト数の方が小さい場合は、バッファーに残りのすべてのバイトが格納され、バッファーは部分的に埋まります。

入力に NULL ポインターを指定すると、メソッドはそのバイトを読み取らずにスキップオーバーします。呼び出しの前の、ストリームから読み取られる残りバイト数が、バッファーの長さより大きい場合、スキップされるバイト数は、バッファーの長さと同じになります。そうでない場合は、残りのすべてのバイトがスキップされます。カーソルは、バイト・メッセージ・ストリームを読み取るために次の位置に残ります。

### length (入力)

バッファーの長さ (バイト単位)

## 戻り値:

バッファーに読み取るバイト数。バッファーが部分的に埋まっている場合は、値はバッファーの長さよりも小さく、読み取るバイトが残っていないことを示します。呼び出し前のストリームに読み取られるバイトが残っていない場合、値は `XMSC_END_OF_STREAM` です。

入力に NULL ポインターを指定すると、メソッドは値を戻しません。

## 例外:

- `XMSEException`
- `MessageNotReadableException`

## *ReadChar* - 文字の読み取り

### インターフェース:

```
Char ReadChar();
```

バイト・メッセージ・ストリームから、次の 2 バイトを文字として読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる文字。

## 例外:

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`

## *ReadDouble* - 倍精度浮動小数点数の読み取り

### インターフェース:

```
Double ReadDouble();
```

バイト・メッセージ・ストリームから、次の 8 バイトを倍精度浮動小数点数として読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる倍精度浮動小数点数。

## 例外:

- `XMSEException`
- `MessageNotReadableException`

- MessageEOFException

*ReadFloat* - 浮動小数点数の読み取り

**インターフェース:**

```
Single ReadFloat();
```

バイト・メッセージ・ストリームから、次の 4 バイトを浮動小数点数として読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる浮動小数点数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 整数の読み取り

**インターフェース:**

```
Int32 ReadInt();
```

バイト・メッセージ・ストリームから、次の 4 バイトを符号付き 32 ビット整数として読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる整数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadLong* - 長整数の読み取り

**インターフェース:**

```
Int64 ReadLong();
```

バイト・メッセージ・ストリームから、次の 8 バイトを符号付き 64 ビット整数として読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる長整数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

## *ReadShort* - 短整数の読み取り

### インターフェース:

```
Int16 ReadShort();
```

バイト・メッセージ・ストリームから、次の 2 バイトを符号付き 16 ビット整数として読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる短整数。

### 例外:

- *XMSEException*
- *MessageNotReadableException*
- *MessageEOFException*

## *ReadByte* - 符号なしバイトの読み取り

### インターフェース:

```
Byte ReadByte();
```

バイト・メッセージ・ストリームから、次のバイトを符号なし 8 ビット整数として読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られるバイト。

### 例外:

- *XMSEException*
- *MessageNotReadableException*
- *MessageEOFException*

## *ReadUnsignedShort* - 符号なし短整数の読み取り

### インターフェース:

```
Int32 ReadUnsignedShort();
```

バイト・メッセージ・ストリームから、次の 2 バイトを符号なし 16 ビット整数として読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる符号なし短整数。

### 例外:

- *XMSEException*
- *MessageNotReadableException*
- *MessageEOFException*

## *ReadUTF* - UTF スtring の読み取り

### インターフェース:

```
String ReadUTF();
```

バイト・メッセージ・ストリームから、UTF-8 でエンコードされた String を読み取ります。

注: ReadUTF() を呼び出す前に、バッファのカーソルがバイト・メッセージ・ストリームの先頭を指すようにしてください。

**パラメーター:**

なし

**戻り値:**

読み取られる文字列をカプセル化している String オブジェクト。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*Reset* - リセット

**インターフェース:**

```
void Reset();
```

メッセージの本体を読み取り専用モードにして、カーソルをバイト・メッセージ・ストリームの先頭に位置変更します。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotReadableException

*WriteBoolean* - ブール値の書き込み

**インターフェース:**

```
void WriteBoolean(Boolean value);
```

バイト・メッセージ・ストリームへブール値を書き込みます。

**パラメーター:**

**value (入力)**

書き込まれるブール値。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteByte* - バイトの書き込み

**インターフェース:**

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

バイト・メッセージ・ストリームへ1バイト書き込みます。

**パラメーター:**

**value (入力)**

書き込まれるバイト。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteBytes* - 複数バイトの書き込み

**インターフェース:**

```
void WriteBytes(Byte[] value);
```

バイト・メッセージ・ストリームへバイトの配列を書き込みます。

**パラメーター:**

**value (入力)**

書き込まれるバイト配列。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteBytes* - 部分的なバイト配列の書き込み

**インターフェース:**

```
void WriteBytes(Byte[] value, int offset, int length);
```

部分的なバイト配列を、指定の長さで定義したとおりにバイト・メッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれるバイト配列。

**offset (入力)**

書き込まれるバイト配列の開始点。

**length (入力)**

書き込むバイト数。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteChar* - 文字の書き込み

**インターフェース:**

```
void WriteChar(Char value);
```

文字を、上位バイトを先にして、2バイトでバイト・メッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれる文字。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteDouble* - 倍精度浮動小数点数の書き込み

**インターフェース:**

```
void WriteDouble(Double value);
```

倍精度浮動小数点数を長整数に変換し、その長整数を、上位バイトを先にして、8バイトでバイト・メッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれる倍精度浮動小数点数。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteFloat* - 浮動小数点数の書き込み

**インターフェース:**

```
void WriteFloat(Single value);
```

浮動小数点数を整数に変換し、その整数を、上位バイトを先にして、4バイトでバイト・メッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれる浮動小数点数。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteInt* - 整数の書き込み

**インターフェース:**

```
void WriteInt(Int32 value);
```

整数を、上位バイトを先にして、4バイトでバイト・メッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれる整数。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

*WriteLong* - 長整数の書き込み

インターフェース:

```
void WriteLong(Int64 value);
```

長整数を、上位バイトを先にして、8 バイトでバイト・メッセージ・ストリームに書き込みます。

パラメーター:

**value (入力)**

書き込まれる長整数。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

*WriteObject* - オブジェクトの書き込み

インターフェース:

```
void WriteObject(Object value);
```

指定したオブジェクトをバイト・メッセージ・ストリームに書き込みます。

パラメーター:

**value (入力)**

書き込まれるオブジェクト。プリミティブ型への参照である必要があります。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

*WriteShort* - 短整数の書き込み

インターフェース:

```
void WriteShort(Int16 value);
```

短整数を、上位バイトを先にして、2 バイトでバイト・メッセージ・ストリームに書き込みます。

パラメーター:

**value (入力)**

書き込まれる短整数。

戻り値:

Void

例外:

- XMSEException

- MessageNotWritableException

WriteUTF - UTF スtringの書き込み

インターフェース:

```
void WriteUTF(String value);
```

バイト・メッセージ・ストリームへ、UTF-8 でエンコードされたストリングを書き込みます。

パラメーター:

**value (入力)**

書き込まれるストリングをカプセル化している String オブジェクト。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

## 継承されたプロパティおよびメソッド

以下のプロパティは、IMessage インターフェースから継承されています。

[JMSCorrelationID](#)、[JMSDeliveryMode](#)、[JMSDestination](#)、[JMSExpiration](#)、[JMSMessageID](#)、[JMSPriority](#)、[JMSRedelivered](#)、[JMSReplyTo](#)、[JMSTimestamp](#)、[JMSType](#)、[Properties](#)

以下のメソッドは、IMessage インターフェースから継承されています。

[clearBody](#)、[clearProperties](#)、[PropertyExists](#)

以下のメソッドは、IPropertyContext インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IConnection

Connection オブジェクトは、アプリケーションからメッセージング・サーバーへのアクティブな接続を表します。

継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnection
```

Connection オブジェクトの XMS 定義プロパティのリストについては、[182 ページの『Connection のプロパティ』](#)を参照してください。

## ドット・ネット プロパティ

ドット・ネット プロパティの要約:

ドット・ネット プロパティ

[ClientID](#)

[ExceptionListener](#)

説明

接続のクライアント ID を取得および設定します。

接続に登録されている例外リスナーを取得し、例外リスナーを接続に登録します。

## ドット・ネット プロパティ

### 説明

#### MetaData

接続のメタデータを取得します。

### ClientID - クライアント ID の取得および設定

#### インターフェース:

```
String ClientID
{
    get;
    set;
}
```

接続のクライアント ID を取得および設定します。

クライアント ID は、ClientID を設定することにより割り当てられることも、管理者が ConnectionFactory で事前に構成することもできます。

クライアント ID は、パブリッシュ/サブスクライブ・ドメイン内の永続サブスクリプションをサポートするためだけに使用され、Point-to-Point ドメインでは無視されます。

アプリケーションが接続のクライアント ID を設定する場合、アプリケーションは、接続の作成の直後に、接続で他の操作を実行する前にこの設定を行う必要があります。アプリケーションが、この時点よりも後に、このクライアント ID の設定を試行すると、呼び出しは例外 `IllegalStateException` をスローします。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

#### 例外:

- `XMSEException`
- `IllegalStateException`
- `InvalidClientIDException`

### ExceptionListener - 例外リスナーの取得および設定

#### インターフェース:

```
ExceptionListener ExceptionListener
{
    get;
    set;
}
```

接続に登録されている例外リスナーを取得し、例外リスナーを接続に登録します。

接続に例外リスナーが登録されていない場合、このメソッドでは `NULL` が戻されます。接続に例外リスナーが既に登録されている場合は、この例外リスナーの代わりに `NULL` を指定すれば、登録を取り消すことができます。

例外リスナーの使用について詳しくは、[50 ページの『ドット・ネットでのメッセージ・リスナーおよび例外リスナー』](#)を参照してください。

#### 例外:

- `XMSEException`

### Metadata - メタデータの取得

#### インターフェース:

```
IConnectionMetaData MetaData
{
    get;
}
```

接続のメタデータを取得します。

## 例外:

- XMSEException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">Close</a>	接続を閉じます。
<a href="#">CreateSession</a>	セッションを作成します。
<a href="#">Start</a>	接続の着信メッセージの配信を開始または再開します。
<a href="#">Stop</a>	接続の着信メッセージの配信を停止します。

### Close - 接続のクローズ

#### インターフェース:

```
void Close();
```

接続を閉じます。

アプリケーションが、既に閉じている接続を閉じようとした場合、呼び出しは無視されます。

#### パラメーター:

なし

#### 戻り値:

Void

## 例外:

- XMSEException

### CreateSession - セッションの作成

#### インターフェース:

```
ISession CreateSession(Boolean transacted,  
                        AcknowledgeMode acknowledgeMode);
```

セッションを作成します。

#### パラメーター:

##### transacted (入力)

値 True は、セッションがトランザクション化されていることを意味します。値 False は、セッションがトランザクション化されていないことを意味します。

ブローカーへのリアルタイム接続の場合、値は False である必要があります。

##### acknowledgeMode (入力)

アプリケーションが受信するメッセージの確認応答の方法を示します。値は、以下の AcknowledgeMode 列挙子のいずれかにする必要があります。

```
AcknowledgeMode.AutoAcknowledge  
AcknowledgeMode.ClientAcknowledge  
AcknowledgeMode.DupsOkAcknowledge
```

ブローカーへのリアルタイム接続の場合、値は AcknowledgeMode.AutoAcknowledge または AcknowledgeMode.DupsOkAcknowledge である必要があります。

セッションがトランザクション化されている場合、このパラメーターは無視されます。肯定応答モードについて詳しくは、28 ページの『メッセージ肯定応答』を参照してください。

**戻り値:**

Session オブジェクト。

**例外:**

- XMSEException

*Start* - 接続の開始

**インターフェース:**

```
void Start();
```

接続の着信メッセージの配信を開始または再開します。接続が既に開始されている場合、呼び出しは無視されます。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException

*Stop* - 接続の停止

**インターフェース:**

```
void Stop();
```

接続の着信メッセージの配信を停止します。接続が既に停止されている場合、呼び出しは無視されます。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IConnectionFactory

アプリケーションは、接続ファクトリーを使用して接続を作成します。

**継承の階層:**

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionFactory
```

ConnectionFactory オブジェクトの XMS 定義プロパティのリストについては、[183 ページの『ConnectionFactory のプロパティ』](#)を参照してください。

**関連概念**

[ConnectionFactories](#) オブジェクトと [Connection](#) オブジェクト

ConnectionFactory オブジェクトには、アプリケーションが Connection オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、Connection オブジェクトを使用して Session オブジェクトを作成します。

#### WebSphere サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

#### IBM WebSphere MQ キュー・マネージャー へのセキュア接続

XMS .NET アプリケーションが、IBM WebSphere MQ キュー・マネージャーとのセキュア接続を確立できるようにするには、関係するプロパティが ConnectionFactory オブジェクトで定義されていることが必要です。

#### WebSphere サービス統合バス メッセージング・エンジンとのセキュア接続

XMS .NET アプリケーションが WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続を確立できるようにするには、関係するプロパティが ConnectionFactory オブジェクトで定義されていることが必要です。

#### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

### 関連タスク

#### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### 関連資料

#### 管理対象 ConnectionFactory オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

## 方法

### メソッドの要約:

メソッド	説明
<u>CreateConnection</u>	デフォルトのプロパティを使用して接続ファクトリーを作成します。
<u>CreateConnection</u>	指定されたユーザー ID を使用して接続を作成します。

*CreateConnection* - 接続ファクトリーの作成 (デフォルト・ユーザー ID を使用)

### インターフェース:

```
ICollection CreateConnection();
```

デフォルトのプロパティを使用して接続ファクトリーを作成します。

IBM WebSphere MQ に接続する場合、接続ファクトリーの XMSC\_USERID プロパティを設定するときは、ログオン・ユーザーの **userid** と一致する必要があります。これらのプロパティを設定しない場合、キュー・マネージャーはデフォルトでログオン・ユーザーの **userid** を使用します。個々のユーザーの追加の接続レベル認証が必要な場合は、IBM WebSphere MQ で構成されるクライアント認証出口を作成できます。

### パラメーター:

なし

### 例外:

- XMSException

CreateConnection - 接続の作成 (指定されたユーザー ID を使用)

インターフェース:

```
IConnection CreateConnection(String userId, String password);
```

指定されたユーザー ID を使用して接続を作成します。

IBM WebSphere MQ に接続する場合、接続ファクトリーの XMSC\_USERID プロパティを設定するときは、ログオン・ユーザーの **userid** と一致している必要があります。これらのプロパティを設定しない場合、キュー・マネージャーはデフォルトでログオン・ユーザーの **userid** を使用します。個々のユーザーの追加の接続レベル認証が必要な場合は、IBM WebSphere MQ で構成されるクライアント認証出口を作成できます。

接続は停止済みモードで作成されます。アプリケーションが **Connection.start()** を呼び出すまで、メッセージは配信されません。

パラメーター:

**userID (入力)**

アプリケーションを認証するときに使用するユーザー ID をカプセル化している String オブジェクト。NULL を指定した場合は、認証のない接続の作成が試行されます。

**password (入力)**

アプリケーションを認証するときに使用するパスワードをカプセル化している String オブジェクト。NULL を指定した場合は、認証のない接続の作成が試行されます。

戻り値:

Connection オブジェクト。

例外:

- XMSEException
- XMS\_X\_SECURITY\_EXCEPTION

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IConnectionMetaData

ConnectionMetaData オブジェクトは、接続に関する情報を提供します。

継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionMetaData
```

ConnectionMetaData オブジェクトの XMS 定義プロパティのリストについては、[189 ページ](#)の『[ConnectionMetaData のプロパティ](#)』を参照してください。

## ドット・ネット プロパティ

プロパティの要約:

メソッド	説明
<a href="#">JMSXPropertyNames</a>	接続でサポートされている JMS 定義メッセージ・プロパティの名前の列挙を戻します。

## JMSXPropertyNames - JMS 定義メッセージ・プロパティの取得

### インターフェース:

```
System.Collections.IEnumerator JMSXPropertyNames
{
    get;
}
```

接続でサポートされている JMS 定義メッセージ・プロパティの名前の列挙を戻します。

JMS 定義メッセージ・プロパティは、ブローカーへのリアルタイム接続ではサポートされていません。

### 例外:

- XMSException

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IDestination

宛先とは、アプリケーションがメッセージを送信する場所、またはアプリケーションがメッセージを受信する場合の送信元、あるいはその両方のことです。

### 継承の階層:

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IDestination
```

Destination オブジェクトの XMS 定義プロパティのリストについては、[189 ページ](#)の『[Destination のプロパティ](#)』を参照してください。

### 関連概念

[ConnectionFactory](#) オブジェクトと [Connection](#) オブジェクト

[ConnectionFactory](#) オブジェクトには、アプリケーションが [Connection](#) オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、[Connection](#) オブジェクトを使用して [Session](#) オブジェクトを作成します。

[WebSphere](#) サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、[WebSphere](#) サービス統合バスに接続できます。

### 宛先

XMS アプリケーションは、送信対象メッセージの宛先と受信対象メッセージの送信元を指定するときに [Destination](#) オブジェクトを使用します。

### 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

### トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティ (複数可) を指定することもできます。

### キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティ (複数可) を指定することもできます。

## 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

## 関連タスク

### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

## 関連資料

### 管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

## ドット・ネット プロパティ

### メソッドの要約:

メソッド	説明
<a href="#">Name</a>	宛先名を取得します。
<a href="#">TypeId</a>	宛先のタイプを取得します。

*Name* - 宛先名の取得

### インターフェース:

```
String Name
{
    get;
}
```

宛先名を取得します。この名前は、キューの名前またはトピックの名前をカプセル化しているストリングです。

### 例外:

- XMSEException

*TypeId* - 宛先タイプの取得

### インターフェース:

```
DestinationType TypeId
{
    get;
}
```

宛先のタイプを取得します。宛先のタイプは、以下の値のいずれかです。

DestinationType.Queue  
DestinationType.Topic

### 例外:

- XMSEException

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## ExceptionHandler

継承の階層:

なし

アプリケーションは例外リスナーを使用して、接続の問題に関する通知を非同期に受信します。

アプリケーションがメッセージを非同期に消費するためにのみ接続を使用し、他の目的では使用しない場合、アプリケーションが接続の問題を確認することができる唯一の方法は、例外リスナーを使用することです。その他の状況では、例外リスナーは、次の XMS への同期呼び出しを待機するよりも迅速に、接続の問題を確認する方法を提供することができます。

## 代行

代行の要約:

代行	説明
<a href="#">ExceptionHandler</a>	アプリケーションに接続の問題を通知します。

*ExceptionHandler* - 例外リスナー

インターフェース:

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

アプリケーションに接続の問題を通知します。

この代行を実装するメソッドは、接続に登録できます。

例外リスナーの使用について詳しくは、[50 ページの『ドット・ネットでのメッセージ・リスナーおよび例外リスナー』](#)を参照してください。

パラメーター:

**exception (入力)**

XMS が作成した例外へのポインター。

戻り値:

Void

## IllegalStateException

継承の階層:

```
IBM.XMS.XMSException
|
+----IBM.XMS.Exception
|
+----IBM.XMS.IllegalStateException
```

XMS がこの例外をスローするのは、アプリケーションがメソッドを呼び出した時刻が誤っているか不適切である場合、あるいは要求した操作に対して XMS が適切な状態でない場合です。

## 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## InitialContext

アプリケーションは、InitialContext オブジェクトを使用して、管理対象オブジェクトのリポジトリから取得したオブジェクト定義によってオブジェクトを作成します。

### 継承の階層:

なし

### 関連概念

#### InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

#### XMS 初期コンテキストの URI フォーマット

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

### 関連タスク

#### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## ドット・ネット プロパティ

### ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<u>Environment</u>	環境を取得します。

#### Environment - 環境の取得

### インターフェース:

```
Hashtable Environment
{
    get;
}
```

環境を取得します。

### 例外:

- 例外は、使用するディレクトリー・サービスに固有のものです。

## コンストラクター

### コンストラクターの要約:

コンストラクター	説明
<u>InitialContext</u>	InitialContext オブジェクトを作成します。

#### InitialContext - 初期コンテキストの作成

### インターフェース:

```
InitialContext(Hashtable env);
```

InitialContext オブジェクトを作成します。

### パラメーター:

管理対象オブジェクトのリポジトリへの接続を確立するために必要な情報は、環境 Hashtable 内のコンストラクターに渡されます。

### 例外:

- XMSEException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">AddToEnvironment</a>	環境へ新規プロパティを追加します。
<a href="#">Close</a>	このコンテキストを閉じます。
<a href="#">Lookup</a>	管理対象オブジェクトのリポジトリから取得したオブジェクト定義によって、オブジェクトを作成します。
<a href="#">RemoveFromEnvironment</a>	環境からプロパティを除去します。

*AddToEnvironment* - 環境への新規プロパティの追加

### インターフェース:

```
Object AddToEnvironment(String propName, Object propVal);
```

環境へ新規プロパティを追加します。

### パラメーター:

#### **propName (入力)**

追加するプロパティの名前をカプセル化している String オブジェクト。

#### **propVal (入力)**

追加するプロパティの値。

### 戻り値:

プロパティの以前の値。

### 例外:

- 例外は、使用するディレクトリー・サービスに固有のものです。

*Close* - このコンテキストのクローズ

### インターフェース:

```
void Close()
```

このコンテキストを閉じます。

### パラメーター:

なし

### 戻り値:

なし

### 例外:

- 例外は、使用するディレクトリー・サービスに固有のものです。

## Lookup - 初期コンテキスト内のオブジェクトの検索

### インターフェース:

```
Object Lookup(String name);
```

管理対象オブジェクトのリポジトリから取得したオブジェクト定義によって、オブジェクトを作成します。

### パラメーター:

#### name (入力)

検索対象の管理対象オブジェクトの名前をカプセル化している String オブジェクト。この名前は、単純な名前でも複雑な名前でも構いません。詳しくは、[65 ページの『管理対象オブジェクトの検索』](#)を参照してください。

### 戻り値:

検索の対象となるオブジェクトのタイプに応じて、IConnectionFactory または IDestination のいずれか。この関数はディレクトリーにアクセスできますが、必要なオブジェクトを検索できないため、NULL が戻ります。

### 例外:

- 例外は、使用するディレクトリー・サービスに固有のものです。

## RemoveFromEnvironment - 環境からのプロパティーの除去

### インターフェース:

```
Object RemoveFromEnvironment(String propName);
```

環境からプロパティーを除去します。

### パラメーター:

#### propName (入力)

除去するプロパティーの名前をカプセル化している String オブジェクト。

### 戻り値:

除去されたオブジェクト。

### 例外:

- 例外は、使用するディレクトリー・サービスに固有のものです。

## InvalidClientIDException

### 継承の階層:

```
IBM.XMS.XMSException
|
+----+ IBM.XMS.XMSException
      |
      +----+ IBM.XMS.InvalidClientIDException
```

XMS がこの例外をスローするのは、アプリケーションが接続のクライアント ID を設定しようとしたが、そのクライアント ID が無効かまたは既に使用中である場合です。

## 継承されたプロパティーおよびメソッド

以下のメソッドは、[XMSException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## InvalidDestinationException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.InvalidDestinationException
```

XMSがこの例外をスローするのは、アプリケーションが無効な宛先を指定している場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## InvalidSelectorException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.InvalidSelectorException
```

XMSがこの例外をスローするのは、無効な構文のメッセージ・セレクター式をアプリケーションで指定した場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## IMapMessage

マップ・メッセージとは、本体が名前と値のペアで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。

継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.IMapMessage
```

アプリケーションが名前と値のペアの値を取得するとき、値はXMSによって別のデータ・タイプに変換される可能性があります。この形式の暗黙の型変換について詳しくは、[79 ページ](#)の『マップ・メッセージ』を参照してください。

### 関連資料

[マップ・メッセージ](#)

マップ・メッセージの本文には、名前値のペアが1組含まれており、それぞれの値には関連付けられたデータ型があります。

## ドット・ネット プロパティ

ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
---------------	----

<a href="#">MapNames</a>	マップ・メッセージの本体に存在する名前の列挙を取得します。
--------------------------	-------------------------------

## MapNames - マップ名の取得

### インターフェース:

```
System.Collections.IEnumerator MapNames
{
    get;
}
```

マップ・メッセージの本体に存在する名前の列挙を取得します。

### 例外:

- XMSEException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">GetBoolean</a>	マップ・メッセージの本体から名前で識別されるブール値を取得します。
<a href="#">GetByte</a>	マップ・メッセージの本体から名前で識別されるバイトを取得します。
<a href="#">GetBytes</a>	マップ・メッセージの本体から名前で識別されるバイトの配列を取得します。
<a href="#">GetChar</a>	マップ・メッセージの本体から名前で識別される文字を取得します。
<a href="#">GetDouble</a>	マップ・メッセージの本体から名前で識別される倍精度浮動小数点数を取得します。
<a href="#">GetFloat</a>	マップ・メッセージの本体から名前で識別される浮動小数点数を取得します。
<a href="#">GetInt</a>	マップ・メッセージの本体から名前で識別される整数を取得します。
<a href="#">GetLong</a>	マップ・メッセージの本体から名前で識別される長整数を取得します。
<a href="#">GetObject</a>	マップ・メッセージの本体から、名前と値のペアの値への参照を取得します。
<a href="#">GetShort</a>	マップ・メッセージの本体から名前で識別される短整数を取得します。
<a href="#">GetString</a>	マップ・メッセージの本体から名前で識別されるストリングを取得します。
<a href="#">ItemExists</a>	マップ・メッセージの本体に、指定した名前が付けられた名前と値のペアが含まれているかどうかをチェックします。
<a href="#">SetBoolean</a>	マップ・メッセージの本体にブール値を設定します。
<a href="#">SetByte</a>	マップ・メッセージの本体にバイトを設定します。
<a href="#">SetBytes</a>	マップ・メッセージの本体にバイトの配列を設定します。
<a href="#">SetChar</a>	マップ・メッセージの本体に2バイト文字を設定します。
<a href="#">SetDouble</a>	マップ・メッセージの本体に倍精度浮動小数点数を設定します。
<a href="#">SetFloat</a>	マップ・メッセージの本体に浮動小数点数を設定します。
<a href="#">SetInt</a>	マップ・メッセージの本体に整数を設定します。
<a href="#">SetLong</a>	マップ・メッセージの本体に長整数を設定します。
<a href="#">SetObject</a>	マップ・メッセージの本体に XMS プリミティブ型の値を設定します。
<a href="#">SetShort</a>	マップ・メッセージの本体に短整数を設定します。
<a href="#">SetString</a>	マップ・メッセージの本体にストリングを設定します。

## GetBoolean - ブール値の取得

### インターフェース:

```
Boolean GetBoolean(String name);
```

マップ・メッセージの本体から名前で識別されるブール値を取得します。

### パラメーター:

#### name (入力)

ブール値を識別する名前をカプセル化している String オブジェクト。

### 戻り値:

マップ・メッセージの本体から検索されたブール値。

### 例外:

- XMSEException

## GetByte - バイトの取得

### インターフェース:

```
Byte    GetByte(String name);  
Int16   GetSignedByte(String name);
```

マップ・メッセージの本体から名前で識別されるバイトを取得します。

### パラメーター:

#### name (入力)

バイトを識別する名前をカプセル化している String オブジェクト。

### 戻り値:

マップ・メッセージの本体から検索されたバイト。バイトにはデータ変換は実行されません。

### 例外:

- XMSEException

## GetBytes - 複数バイトの取得

### インターフェース:

```
Byte[]  GetBytes(String name);
```

マップ・メッセージの本体から名前で識別されるバイトの配列を取得します。

### パラメーター:

#### name (入力)

バイトの配列を識別する名前をカプセル化している String オブジェクト。

### 戻り値:

配列のバイト数。

### 例外:

- XMSEException

## GetChar - 文字の取得

### インターフェース:

```
Char    GetChar(String name);
```

マップ・メッセージの本体から名前で識別される文字を取得します。

**パラメーター:**

**name (入力)**

文字を識別する名前をカプセル化している String オブジェクト。

**戻り値:**

マップ・メッセージの本体から検索された文字。

**例外:**

- XMSEException

*GetDouble* - 倍精度浮動小数点数の取得

**インターフェース:**

```
Double GetDouble(String name);
```

マップ・メッセージの本体から名前前で識別される倍精度浮動小数点数を取得します。

**パラメーター:**

**name (入力)**

倍精度浮動小数点数を識別する名前をカプセル化している String オブジェクト。

**戻り値:**

マップ・メッセージの本体から検索された倍精度浮動小数点数。

**例外:**

- XMSEException

*GetFloat* - 浮動小数点数の取得

**インターフェース:**

```
Single GetFloat(String name);
```

マップ・メッセージの本体から名前前で識別される浮動小数点数を取得します。

**パラメーター:**

**name (入力)**

浮動小数点数を識別する名前をカプセル化している String オブジェクト。

**戻り値:**

マップ・メッセージの本体から検索された浮動小数点数。

**例外:**

- XMSEException

*GetInt* - 整数の取得

**インターフェース:**

```
Int32 GetInt(String name);
```

マップ・メッセージの本体から名前前で識別される整数を取得します。

**パラメーター:**

**name (入力)**

整数を識別する名前をカプセル化している String オブジェクト。

**戻り値:**

マップ・メッセージの本体から検索された整数。

**例外:**

- XMSEException

## GetLong - 長整数の取得

### インターフェース:

```
Int64 GetLong(String name);
```

マップ・メッセージの本体から名前でも識別される長整数を取得します。

### パラメーター:

#### name (入力)

長整数を識別する名前をカプセル化している String オブジェクト。

### 戻り値:

マップ・メッセージの本体から検索された長整数。

### 例外:

- XMSEException

## GetObject - オブジェクトの取得

### インターフェース:

```
Object GetObject(String name);
```

マップ・メッセージの本体から、名前と値のペアの値への参照を取得します。名前と値のペアは、名前でも識別されます。

### パラメーター:

#### name (入力)

名前と値のペアの名前をカプセル化している String オブジェクト。

### 戻り値:

値。以下のオブジェクト・タイプのいずれかです。

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

### 例外:

- XMSEException

## GetShort - 短整数の取得

### インターフェース:

```
Int16 GetShort(String name);
```

マップ・メッセージの本体から名前でも識別される短整数を取得します。

### パラメーター:

#### name (入力)

短整数を識別する名前をカプセル化している String オブジェクト。

### 戻り値:

マップ・メッセージの本体から検索された短整数。

#### 例外:

- XMSEException

*GetString* - スtringの取得

#### インターフェース:

```
String GetString(String name);
```

マップ・メッセージの本体から名前と識別されるStringを取得します。

#### パラメーター:

##### name (入力)

マップ・メッセージの本体のStringを識別する名前をカプセル化しているStringオブジェクト。

#### 戻り値:

マップ・メッセージの本体から取り出したStringをカプセル化しているStringオブジェクト。データ変換が必要な場合、この値は、変換後のStringになります。

#### 例外:

- XMSEException

*ItemExists* - 名前と値のペアの存在のチェック

#### インターフェース:

```
Boolean ItemExists(String name);
```

マップ・メッセージの本体に、指定した名前が付けられた名前と値のペアが含まれているかどうかをチェックします。

#### パラメーター:

##### name (入力)

名前と値のペアの名前をカプセル化しているStringオブジェクト。

#### 戻り値:

- マップ・メッセージの本体に、指定された名前が付けられた名前と値のペアが含まれている場合は、Trueです。
- マップ・メッセージの本体に、指定された名前が付けられた名前と値のペアが含まれていない場合は、Falseです。

#### 例外:

- XMSEException

*SetBoolean* - ブール値の設定

#### インターフェース:

```
void SetBoolean(String name, Boolean value);
```

マップ・メッセージの本体にブール値を設定します。

#### パラメーター:

##### name (入力)

マップ・メッセージの本体に存在するブール値を識別するための名前をカプセル化しているStringオブジェクト。

##### value (入力)

設定されるブール値。

**戻り値:**

Void

**例外:**

- XMSEException

**SetByte - バイトの設定****インターフェース:**

```
void SetByte(String name, Byte value);  
void SetSignedByte(String name, Int16 value);
```

マップ・メッセージの本体にバイトを設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在するバイトを識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定されるバイト。

**戻り値:**

Void

**例外:**

- XMSEException

**SetBytes - 複数バイトの設定****インターフェース:**

```
void SetBytes(String name, Byte[] value);
```

マップ・メッセージの本体にバイトの配列を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在するバイトの配列を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定されるバイト配列。

**戻り値:**

Void

**例外:**

- XMSEException

**SetChar - 文字の設定****インターフェース:**

```
void SetChar(String name, Char value);
```

マップ・メッセージの本体に 2 バイト文字を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する文字を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される文字。

**戻り値:**

Void

**例外:**

- XMSEException

*SetDouble* - 倍精度浮動小数点数の設定

**インターフェース:**

```
void SetDouble(String name, Double value);
```

マップ・メッセージの本体に倍精度浮動小数点数を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する倍精度浮動小数点数を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される倍精度浮動小数点数。

**戻り値:**

Void

**例外:**

- XMSEException

*SetFloat* - 浮動小数点数の設定

**インターフェース:**

```
void SetFloat(String name, Single value);
```

マップ・メッセージの本体に浮動小数点数を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する浮動小数点数を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される浮動小数点数。

**戻り値:**

Void

**例外:**

- XMSEException

*SetInt* - 整数の設定

**インターフェース:**

```
void SetInt(String name, Int32 value);
```

マップ・メッセージの本体に整数を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する整数を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される整数。

**戻り値:**

Void

**例外:**

- XMSEException

**SetLong - 長整数の設定****インターフェース:**

```
void SetLong(String name, Int64 value);
```

マップ・メッセージの本体に長整数を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する長整数を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される長整数。

**戻り値:**

Void

**例外:**

- XMSEException

**SetObject - オブジェクトの設定****インターフェース:**

```
void SetObject(String name, Object value);
```

マップ・メッセージの本体に XMS プリミティブ型の値を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する値を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される値を含むバイトの配列。

**戻り値:**

Void

**例外:**

- XMSEException

**SetShort - 短整数の設定****インターフェース:**

```
void SetShort(String name, Int16 value);
```

マップ・メッセージの本体に短整数を設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在する短整数を識別するための名前をカプセル化している String オブジェクト。

**value (入力)**

設定される短整数。

**戻り値:**

Void

**例外:**

- XMSEException

SetString - スtringの設定

**インターフェース:**

```
void SetString(String name, String value);
```

マップ・メッセージの本体にStringを設定します。

**パラメーター:****name (入力)**

マップ・メッセージの本体に存在するStringを識別するための名前をカプセル化しているStringオブジェクト。

**value (入力)**

設定されるStringをカプセル化しているStringオブジェクト。

**戻り値:**

Void

**例外:**

- XMSEException

**継承されたプロパティおよびメソッド**

以下のプロパティは、IMessage インターフェースから継承されています。

[JMSCorrelationID](#)、[JMSDeliveryMode](#)、[JMSDestination](#)、[JMSExpiration](#)、[JMSMessageID](#)、[JMSPriority](#)、[JMSRedelivered](#)、[JMSReplyTo](#)、[JMSTimestamp](#)、[JMSType](#)、[Properties](#)

以下のメソッドは、IMessage インターフェースから継承されています。

[clearBody](#)、[clearProperties](#)、[PropertyExists](#)

以下のメソッドは、IPropertyContext インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

**IMessage**

メッセージ・オブジェクトは、アプリケーションが送信または受信するメッセージを表します。IMessage は、IMapMessage などの Message クラスのスーパークラスです。

**継承の階層:**

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
```

Message オブジェクトの JMS メッセージ・ヘッダー・フィールドのリストについては、72 ページの『[XMS メッセージのヘッダー・フィールド](#)』を参照してください。Message オブジェクトの JMS 定義プロパティのリストについては、74 ページの『[メッセージの JMS 定義プロパティ](#)』を参照してください。

Message オブジェクトの IBM 定義プロパティのリストについては、75 ページの『[メッセージの IBM 定](#)』

『[メッセージプロパティ](#)』を参照してください。Message オブジェクトの JMS\_IBM\_MQMD\* プロパティのリストについては、[194 ページ](#)の『[JMS\\_IBM\\_MQMD\\* プロパティ](#)』を参照してください。

メッセージは、ガーベッジ・コレクターによって削除されます。メッセージが削除されると、これによってメッセージが使用していたリソースが解放されます。

## 関連資料

### XMS メッセージ

この節章では、XMS メッセージの構造とコンテンツについて説明するとともに、アプリケーションによる XMS メッセージの処理方法について説明します。

## ドット・ネット プロパティ

### ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<a href="#">JMSCorrelationID</a>	メッセージの関連 ID を String オブジェクトとして取得および設定します。
<a href="#">JMSDeliveryMode</a>	メッセージの送達モードを取得して設定します。
<a href="#">JMSDestination</a>	メッセージの宛先を取得して設定します。
<a href="#">JMSExpiration</a>	メッセージの有効期限切れ時刻を取得および設定します。
<a href="#">JMSMessageID</a>	メッセージのメッセージ ID を、このメッセージ ID をカプセル化している String オブジェクトとして取得し、設定します。
<a href="#">JMSPriority</a>	メッセージの優先順位を取得して設定します。
<a href="#">JMSRedelivered</a>	メッセージが再配信されるかどうかの標識を取得し、メッセージが再配信されるかどうかを示します。
<a href="#">JMSReplyTo</a>	メッセージに対する応答が送信される宛先を取得および設定します。
<a href="#">JMSTimestamp</a>	メッセージが送信された時刻を取得して設定します。
<a href="#">JMSType</a>	メッセージのタイプを取得および設定します。
<a href="#">PropertyNames</a>	メッセージの名前プロパティの列挙を取得します。

### GetJMSCorrelationID - JMSCorrelationID の取得および設定

#### インターフェース:

```
String JMSCorrelationID
{
    get;
    set;
}
```

メッセージの関連 ID を String オブジェクトとして取得および設定します。

#### 例外:

- XMSException

### JMSDeliveryMode - JMSDeliveryMode の取得および設定

#### インターフェース:

```
DeliveryMode JMSDeliveryMode
{
    get;
    set;
}
```

メッセージの送達モードを取得して設定します。

メッセージの送達モードは、以下の値のいずれかです。

```
DeliveryMode.Persistent  
DeliveryMode.NonPersistent
```

新規に作成されて送信されていないメッセージの場合、送達モードは `DeliveryMode.Persistent` です。ただし、送達モードが `DeliveryMode.NonPersistent` であるブローカーへのリアルタイム接続の場合を除きます。受信されたメッセージについては、受信側アプリケーションが `JMSDeliveryMode` を設定して送達モードを変更していない限り、このメソッドでは、メッセージ送信時に `IMessageProducer.send()` 呼び出しによって設定された送達モードが戻されます。

**例外:**

- `XMSException`

*JMSDestination* - `JMSDestination` の取得および設定

**インターフェース:**

```
IDestination JMSDestination  
{  
    get;  
    set;  
}
```

メッセージの宛先を取得して設定します。

宛先は、メッセージの送信時に `IMessageProducer.send()` 呼び出しによって設定されます。`JMSDestination` の値は無視されます。ただし、`JMSDestination` を使用して、受信されたメッセージの宛先を変更することができます。

未送信の新規作成メッセージの場合、送信側アプリケーションが `JMSDestination` を設定して宛先を設定しない限り、メソッドはヌルの `Destination` オブジェクトを戻します。受信されたメッセージについては、受信側アプリケーションが `JMSDestination` を設定して宛先を変更していない限り、このメソッドでは、メッセージ送信時に `IMessageProducer.send()` 呼び出しによって設定された宛先の `Destination` オブジェクトが戻されます。

**例外:**

- `XMSException`

*JMSExpiration* - `JMSExpiration` の取得および設定

**インターフェース:**

```
Int64 JMSExpiration  
{  
    get;  
    set;  
}
```

メッセージの有効期限切れ時刻を取得および設定します。

有効期限切れ時刻は、メッセージの送信時に `IMessageProducer.send()` 呼び出しによって設定されます。その値は、送信側アプリケーションが指定した存続時間を、メッセージの送信時刻に加算して計算されます。有効期限切れ時刻は、1970年1月1日 00:00:00 GMT からのミリ秒で表されます。

新規に作成されて送信されていないメッセージの場合、送信側アプリケーションが `JMSExpiration` を設定して異なる有効期限切れ時刻を設定していない限り、有効期限切れ時刻は 0 です。受信されたメッセージについては、受信側アプリケーションが `JMSExpiration` を設定して有効期限切れ時刻を変更していない限り、このメソッドでは、メッセージ送信時に `IMessageProducer.send()` 呼び出しによって設定された有効期限切れ時刻が戻されます。

存続時間が 0 の場合、`IMessageProducer.send()` 呼び出しでは、有効期限切れ時刻が 0 に設定されますが、これはメッセージの有効期限がないことを示します。

XMS は、有効期限が切れたメッセージを廃棄し、アプリケーションに配信しません。

#### 例外:

- XMSEException

#### JMSMessageID - JMSMessageID の取得および設定

##### インターフェース:

```
String JMSMessageID
{
    get;
    set;
}
```

メッセージのメッセージ ID を、このメッセージ ID をカプセル化している String オブジェクトとして取得し、設定します。

メッセージ ID は、メッセージの送信時に IMessageProducer.send() 呼び出しによって設定されます。受信されたメッセージについては、受信側アプリケーションが JMSMessageID を設定してメッセージ ID を変更していない限り、このメソッドでは、メッセージ送信時に IMessageProducer.send() 呼び出しによって設定されたメッセージ ID が戻されます。

メッセージにメッセージ ID がない場合、このメソッドは NULL を戻します。

#### 例外:

- XMSEException

#### JMSPriority - JMSPriority の取得および設定

##### インターフェース:

```
Int32 JMSPriority
{
    get;
    set;
}
```

メッセージの優先順位を取得して設定します。

優先順位は、メッセージの送信時に IMessageProducer.send() 呼び出しによって設定されます。値は 0 (最低優先順位) から 9 (最高優先順位) までの整数です。

新規に作成されて送信されていないメッセージの場合、送信側アプリケーションが JMSPriority を設定して異なる優先順位を設定していない限り、優先順位は 4 です。受信されたメッセージについては、受信側アプリケーションが JMSPriority を設定して優先順位を変更していない限り、このメソッドでは、メッセージ送信時に IMessageProducer.send() 呼び出しによって設定された優先順位が戻されます。

#### 例外:

- XMSEException

#### JMSRedelivered - JMSRedelivered の取得および設定

##### インターフェース:

```
Boolean JMSRedelivered
{
    get;
    set;
}
```

メッセージが再配信されるかどうかの標識を取得し、メッセージが再配信されるかどうかを示します。標識は、メッセージの受信時に IMessageConsumer.receive() 呼び出しによって設定されます。

このプロパティの値は、以下のとおりです。

- メッセージが再配信される場合は、True です。
- メッセージが再配信されない場合は、False です。

ブローカーへのリアルタイム接続の場合、常に値は `False` です。

メッセージの送信前に `JMSRedelivered` が設定した再配信の標識は、メッセージの送信時の `IMessageProducer.send()` 呼び出しでは無視され、メッセージ受信時の `IMessageConsumer.receive()` 呼び出しでは無視されて置き換えられます。ただし、`JMSRedelivered` を使用して、受信したメッセージの標識を変更することもできます。

#### 例外:

- `XMSEException`

#### `JMSReplyTo` - `JMSReplyTo` の取得および設定

##### インターフェース:

```
IDestination JMSReplyTo
{
    get;
    set;
}
```

メッセージに対する応答が送信される宛先を取得および設定します。

このプロパティの値は、メッセージに対する応答が送信される宛先の `Destination` オブジェクトです。`Destination` オブジェクトがヌルの場合は、応答を想定していないという意味です。

#### 例外:

- `XMSEException`

#### `JMSTimestamp` - `JMSTimestamp` の取得および設定

##### インターフェース:

```
Int64 JMSTimestamp
{
    get;
    set;
}
```

メッセージが送信された時刻を取得して設定します。

タイム・スタンプは、メッセージの送信時に `IMessageProducer.send()` 呼び出しによって設定され、1970年1月1日 00:00:00 GMT からのミリ秒で表されます。

新規に作成されて送信されていないメッセージの場合、送信側アプリケーションが `JMSTimestamp` を設定して異なるタイム・スタンプを設定していない限り、タイム・スタンプは `0` です。受信されたメッセージについては、受信側アプリケーションが `JMSTimestamp` を設定してタイム・スタンプを変更していない限り、このメソッドでは、メッセージ送信時に `IMessageProducer.send()` 呼び出しによって設定されたタイム・スタンプが戻されます。

#### 例外:

- `XMSEException`

#### 注:

1. タイム・スタンプが未定義の場合、このメソッドは `0` を戻しますが例外はスローしません。

#### `JMSType` - `JMSType` の取得および設定

##### インターフェース:

```
String JMSType
{
    get;
    set;
}
```

メッセージのタイプを取得および設定します。

JMSType の値は、メッセージのタイプをカプセル化している文字列です。データ変換が必要な場合、この値は、変換後のタイプになります。

**例外:**

- XMSEException

PropertyNames - プロパティの取得

**インターフェース:**

```
System.Collections.IEnumerator PropertyNames
{
    get;
}
```

メッセージの名前プロパティの列挙を取得します。

**例外:**

- XMSEException

## 方法

**メソッドの要約:**

メソッド	説明
<a href="#">Acknowledge</a>	このメッセージと、セッションが以前に受信して、まだ応答していないすべてのメッセージに応答します。
<a href="#">ClearBody</a>	メッセージの本体をクリアします。
<a href="#">ClearProperties</a>	メッセージのプロパティをクリアします。
<a href="#">PropertyExists</a>	メッセージに、指定された名前のプロパティがあるかどうかを検査します。

Acknowledge - 応答

**インターフェース:**

```
void Acknowledge();
```

このメッセージと、セッションが以前に受信して、まだ応答していないすべてのメッセージに応答します。

セッションの応答モードが AcknowledgeMode.ClientAcknowledge である場合、アプリケーションはこのメソッドを呼び出すことができます。セッションがその他の応答モードであるか、セッションが処理中である場合は、このメソッドの呼び出しは無視されます。

受信されたが、応答されていないメッセージは、再配信される可能性があります。

メッセージの応答について詳しくは、[28 ページの『メッセージ肯定応答』](#)を参照してください。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException
- IllegalStateException

## ClearBody - 本体のクリア

### インターフェース:

```
void ClearBody();
```

メッセージの本体をクリアします。ヘッダー・フィールドおよびメッセージ・プロパティはクリアされません。

アプリケーションがメッセージ本体をクリアすると、本体は、新規に作成されたメッセージ内の空の本体と同じ状態になります。新規に作成されたメッセージ内の空の本体の状態は、メッセージ本体のタイプによって異なります。詳しくは、[76 ページの『XMS メッセージの本文』](#)を参照してください。

アプリケーションは、本体の状態にかかわらず、いつでもメッセージ本体をクリアできます。メッセージ本体が読み取り専用の場合、アプリケーションが本体に書き込むことができる唯一の方法は、まずアプリケーションが本体をクリアすることです。

### パラメーター:

なし

### 戻り値:

Void

### 例外:

- XMSEException

## ClearProperties - プロパティのクリア

### インターフェース:

```
void ClearProperties();
```

メッセージのプロパティをクリアします。ヘッダー・フィールドおよびメッセージ本体はクリアされません。

アプリケーションがメッセージのプロパティをクリアした場合、プロパティは読み書き可能になります。

アプリケーションは、プロパティの状態にかかわらず、いつでもメッセージのプロパティをクリアできます。メッセージのプロパティが読み取り専用の場合、プロパティを書き込み可能にすることができる唯一の方法は、アプリケーションがまずプロパティをクリアすることです。

### パラメーター:

なし

### 戻り値:

Void

### 例外:

- XMSEException

## PropertyExists - プロパティの存在の検査

### インターフェース:

```
Boolean PropertyExists(String propertyName);
```

メッセージに、指定された名前のプロパティがあるかどうかを検査します。

### パラメーター:

#### **propertyName (入力)**

プロパティの名前をカプセル化している String オブジェクト。

### 戻り値:

- メッセージに、指定された名前のプロパティがある場合は、True です。

- メッセージに、指定された名前のプロパティがない場合は、False です。

**例外:**

- XMSEException

**継承されたプロパティおよびメソッド**

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

**IMessageConsumer**

アプリケーションは、メッセージ・コンシューマーを使用して、宛先に送信されたメッセージを受信します。

**継承の階層:**



MessageConsumer オブジェクトの XMS 定義プロパティのリストについては、[198 ページの『MessageConsumer のプロパティ』](#)を参照してください。

**ドット・ネット プロパティ**

**ドット・ネット プロパティの要約:**

ドット・ネット プロパティ	説明
<a href="#">MessageListener</a>	メッセージ・コンシューマーに登録されているメッセージ・リスナーを取得し、メッセージ・リスナーをメッセージ・コンシューマーに登録します。
<a href="#">MessageSelector</a>	メッセージ・コンシューマーのメッセージ・セレクターを取得します。

*MessageListener* - メッセージ・リスナーの取得および設定

**インターフェース:**

```

MessageListener MessageListener
{
    get;
    set;
}

```

メッセージ・コンシューマーに登録されているメッセージ・リスナーを取得し、メッセージ・リスナーをメッセージ・コンシューマーに登録します。

メッセージ・コンシューマーにメッセージ・リスナーが登録されていない場合、MessageListener は NULL です。メッセージ・コンシューマーにメッセージ・リスナーが既に登録されている場合、代わりに NULL を指定することによって、登録を取り消すことができます。

メッセージ・リスナーの使用について詳しくは、[50 ページの『ドット・ネットでのメッセージ・リスナーおよび例外リスナー』](#)を参照してください。

**例外:**

- XMSEException

## MessageSelector - メッセージ・セレクターの取得

### インターフェース:

```
String MessageSelector
{
    get;
}
```

メッセージ・コンシューマーのメッセージ・セレクターを取得します。戻り値は、メッセージ・セレクター式をカプセル化している String オブジェクトです。データ変換が必要な場合、この値は、変換後のメッセージ・セレクター式になります。メッセージ・コンシューマーにメッセージ・セレクターが存在しない場合、MessageSelector の値はヌルの String オブジェクトです。

### 例外:

- XMSEException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">Close</a>	メッセージ・コンシューマーを閉じます。
<a href="#">Receive</a>	メッセージ・コンシューマーの次のメッセージを受信します。この呼び出しでは、無期限にメッセージを待機し続けるか、メッセージ・コンシューマーがクローズされるまで待機します。
<a href="#">Receive</a>	メッセージ・コンシューマーの次のメッセージを受信します。この呼び出しは、指定の期間だけメッセージを待機するか、メッセージ・コンシューマーがクローズされるまで待機します。
<a href="#">ReceiveNoWait</a>	メッセージ・コンシューマーの次のメッセージが即時に受信可能である場合に、そのメッセージを受け取ります。

### Close - メッセージ・コンシューマーのクローズ

#### インターフェース:

```
void Close();
```

メッセージ・コンシューマーを閉じます。

アプリケーションが、既に閉じているメッセージ・コンシューマーを閉じようとした場合、呼び出しは無視されます。

#### パラメーター:

なし

#### 戻り値:

Void

### 例外:

- XMSEException

### Receive - 受信

#### インターフェース:

```
IMessage Receive();
```

メッセージ・コンシューマーの次のメッセージを受信します。この呼び出しでは、無期限にメッセージを待機し続けるか、メッセージ・コンシューマーがクローズされるまで待機します。

**パラメーター:**

なし

**戻り値:**

Message オブジェクトへのポインター。呼び出しがメッセージを待機している間にメッセージ・コンシューマーを閉じると、メソッドは、ヌルの Message オブジェクトを指すポインターを戻します。

**例外:**

- XMSEException

*Receive* - 受信 (待機間隔あり)

**インターフェース:**

```
IMessage Receive(Int64 delay);
```

メッセージ・コンシューマーの次のメッセージを受信します。この呼び出しは、指定の期間だけメッセージを待機するか、メッセージ・コンシューマーがクローズされるまで待機します。

**パラメーター:**

**delay (入力)**

呼び出しがメッセージを待機する時間 (ミリ秒単位)。待機間隔を 0 と指定した場合、呼び出しは無期限にメッセージを待機します。

**戻り値:**

Message オブジェクトへのポインター。待機間隔の間にメッセージが到着しなかった場合や、呼び出しがメッセージを待機している間にメッセージ・コンシューマーを閉じた場合、メソッドは、ヌルの Message オブジェクトを指すポインターを戻しますが、例外はスローしません。

**例外:**

- XMSEException

*ReceiveNoWait* - 待機なしの受信

**インターフェース:**

```
IMessage ReceiveNoWait();
```

メッセージ・コンシューマーの次のメッセージが即時に受信可能である場合に、そのメッセージを受け取ります。

**パラメーター:**

なし

**戻り値:**

Message オブジェクトへのポインター。即時に有効なメッセージがない場合、メソッドは、ヌルの Message オブジェクトを指すポインターを戻します。

**例外:**

- XMSEException

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## MessageEOFException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageEOFException
```

XMS がこの例外をスローするのは、アプリケーションがバイト・メッセージの本体を読み取っているときに、XMS がバイト・メッセージ・ストリームの終端を検出した場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## MessageFormatException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageFormatException
```

XMS がこの例外をスローするのは、XMS が無効なフォーマットのメッセージを検出した場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## IMessageListener (代行)

継承の階層:

なし

アプリケーションは、メッセージ・リスナーを使用して、メッセージを非同期に受信します。

### 代行

メソッドの要約:

メソッド	説明
<a href="#">MessageListener</a>	メッセージを非同期にメッセージ・コンシューマーに配信します。

*MessageListener* - メッセージ・リスナー

インターフェース:

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

メッセージを非同期にメッセージ・コンシューマーに配信します。

この代行を実装するメソッドは、接続に登録できます。

メッセージ・リスナーの使用について詳しくは、50 ページの『ドット・ネットでのメッセージ・リスナーおよび例外リスナー』を参照してください。

パラメーター:

**mesg (入力)**

Message オブジェクト。

戻り値:

Void

## MessageNotReadableException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageNotReadableException
```

XMS がこの例外をスローするのは、書き込み専用になっているメッセージの本体をアプリケーションが読み取ろうとした場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## MessageNotWritableException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageNotWritableException
```

XMS がこの例外をスローするのは、読み取り専用になっているメッセージの本体にアプリケーションが書き込もうとした場合です。

### 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## IMessageProducer

アプリケーションは、メッセージ・プロデューサーを使用して、メッセージを宛先に送信します。

継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageProducer
```

MessageProducer オブジェクトの XMS 定義プロパティのリストについては、[198 ページ](#)の『[MessageProducer のプロパティ](#)』を参照してください。

## ドット・ネット プロパティ

### ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<a href="#">DeliveryMode</a>	メッセージ・プロデューサーによって送信されるメッセージのデフォルト送達モードを取得および設定します。
<a href="#">Destination</a>	メッセージ・プロデューサーの宛先を取得します。
<a href="#">DisableMsgID</a>	受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要があるかどうかの標識を取得し、受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要があるかどうかを示します。
<a href="#">DisableMsgTS</a>	受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要があるかどうかの標識を取得し、受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要があるかどうかを示します。
<a href="#">Priority</a>	メッセージ・プロデューサーによって送信されるメッセージのデフォルト優先順位を取得および設定します。
<a href="#">TimeToLive</a>	メッセージが有効期限切れになるまでのデフォルトの時間の長さを取得して設定します。

### *DeliveryMode* - デフォルト送達モードの取得および設定

#### インターフェース:

```
DeliveryMode DeliveryMode
{
    get;
    set;
}
```

メッセージ・プロデューサーによって送信されるメッセージのデフォルト送達モードを取得および設定します。

デフォルトの送達モードは、以下の値のいずれかです。

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

ブローカーへのリアルタイム接続の場合、値は `DeliveryMode.NonPersistent` である必要があります。

デフォルト値は `DeliveryMode.Persistent` です。ただし、デフォルト値が `DeliveryMode.NonPersistent` となるブローカーへのリアルタイム接続の場合を除きます。

#### 例外:

- `XMSException`

### *Destination* - 宛先の取得

#### インターフェース:

```
IDestination Destination
{
    get;
}
```

メッセージ・プロデューサーの宛先を取得します。

**パラメーター:**

なし

**戻り値:**

Destination オブジェクト。メッセージ・プロデューサーに宛先が存在しない場合、このメソッドはヌルの Destination オブジェクトを戻します。

**例外:**

- XMSEException

*DisableMsgID* - メッセージ ID の使用不可化フラグの取得および設定

**インターフェース:**

```
Boolean DisableMessageID
{
    get;
    set;
}
```

受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要があるかどうかの標識を取得し、受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要があるかどうかを示します。

キュー・マネージャーへの接続またはブローカーへのリアルタイム接続では、このフラグは無視されます。サービス統合バスへの接続では、フラグが尊重されます。

DisabledMsgID の値は、以下のとおりです。

- 受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要がない場合は、True です。
- 受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにメッセージ ID が含まれている必要がある場合は、False です。

**例外:**

- XMSEException

*DisableMsgTS* - タイム・スタンプの使用不可化フラグの取得および設定

**インターフェース:**

```
Boolean DisableMessageTimestamp
{
    get;
    set;
}
```

受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要があるかどうかの標識を取得し、受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要があるかどうかを示します。

ブローカーへのリアルタイム接続では、このフラグは無視されます。キュー・マネージャーへの接続またはサービス統合バスへの接続では、フラグは尊重されます。

DisableMsgTS の値は、以下のとおりです。

- 受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要がない場合は、True です。
- 受信側アプリケーションにとって、メッセージ・プロデューサーにより送信されるメッセージにタイム・スタンプが含まれている必要がある場合は、False です。

**戻り値:**

## 例外:

- XMSEException

*Priority* - デフォルト優先順位の取得および設定

## インターフェース:

```
Int32 Priority
{
    get;
    set;
}
```

メッセージ・プロデューサーによって送信されるメッセージのデフォルト優先順位を取得および設定します。

デフォルトのメッセージ優先順位の値は、0 (最低優先順位) から 9 (最高優先順位) までの整数です。

ブローカーへのリアルタイム接続では、メッセージの優先順位は無視されます。

## 例外:

- XMSEException

*TimeToLive* - デフォルト存続時間の取得および設定

## インターフェース:

```
Int64 TimeToLive
{
    get;
    set;
}
```

メッセージが有効期限切れになるまでのデフォルトの時間の長さを取得して設定します。

時間は、メッセージ・プロデューサーがメッセージを送信した時刻とデフォルトの存続時間を基に、ミリ秒単位で測定します。値 0 は、メッセージの有効期限がないことを意味します。

ブローカーへのリアルタイム接続の場合、この値は常に 0 です。

## 例外:

- XMSEException

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">Close</a>	メッセージ・プロデューサーを閉じます。
<a href="#">Send</a>	メッセージ・プロデューサーが作成されたときに指定された宛先にメッセージを送信します。メッセージ・プロデューサーのデフォルト送達モード、優先順位、および存続時間を使用してメッセージを送信します。
<a href="#">Send</a>	メッセージ・プロデューサーが作成されたときに指定された宛先にメッセージを送信します。指定された送達モード、優先順位、および存続時間を使用してメッセージを送信します。
<a href="#">Send</a>	メッセージ・プロデューサーが作成されたときに宛先が指定されていないメッセージ・プロデューサーを使用している場合は、指定された宛先にメッセージを送信します。メッセージ・プロデューサーのデフォルト送達モード、優先順位、および存続時間を使用してメッセージを送信します。

## メソッド

## 説明

### Send

メッセージ・プロデューサーが作成されたときに宛先が指定されていないメッセージ・プロデューサーを使用している場合は、指定された宛先にメッセージを送信します。指定された送達モード、優先順位、および存続時間を使用してメッセージを送信します。

*Close* - メッセージ・プロデューサーのクローズ

### インターフェース:

```
void Close();
```

メッセージ・プロデューサーを閉じます。

アプリケーションが、既に閉じているメッセージ・プロデューサーを閉じようとした場合、呼び出しは無視されます。

### パラメーター:

なし

### 戻り値:

Void

### 例外:

- XMSEException

*Send* - 送信

### インターフェース:

```
void Send(IMessage msg) ;
```

メッセージ・プロデューサーが作成されたときに指定された宛先にメッセージを送信します。メッセージ・プロデューサーのデフォルト送達モード、優先順位、および存続時間を使用してメッセージを送信します。

### パラメーター:

#### **msg (入力)**

Message オブジェクト。

### 戻り値:

Void

### 例外:

- XMSEException
- MessageFormatException
- InvalidDestinationException

*Send* - 送信 (送達モード、優先順位、および存続時間を指定)

### インターフェース:

```
void Send(IMessage msg,  
          DeliveryMode deliveryMode,  
          Int32 priority,  
          Int64 timeToLive);
```

メッセージ・プロデューサーが作成されたときに指定された宛先にメッセージを送信します。指定された送達モード、優先順位、および存続時間を使用してメッセージを送信します。

### パラメーター:

#### **msg (入力)**

Message オブジェクト。

### **deliveryMode (入力)**

メッセージの送達モード。以下の値のいずれかでなければなりません。

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

ブローカーへのリアルタイム接続の場合、値は DeliveryMode.NonPersistent である必要があります。

### **priority (入力)**

メッセージの優先順位。値は 0 (最低優先順位) から 9 (最高優先順位) までの整数です。ブローカーへのリアルタイム接続では、値は無視されます。

### **timeToLive (入力)**

メッセージの存続時間 (ミリ秒単位)。値 0 は、メッセージの有効期限がないことを意味します。ブローカーへのリアルタイム接続の場合、値は 0 である必要があります。

#### 戻り値:

Void

#### 例外:

- XMSEException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

送信 (指定された宛先へ)

#### インターフェース:

```
void Send(IDestination dest, IMessage msg) ;
```

メッセージ・プロデューサーが作成されたときに宛先が指定されていないメッセージ・プロデューサーを使用している場合は、指定された宛先にメッセージを送信します。メッセージ・プロデューサーのデフォルト送達モード、優先順位、および存続時間を使用してメッセージを送信します。

通常、メッセージ・プロデューサーの作成時には宛先を指定しますが、宛先を指定しない場合、メッセージの送信ごとに宛先を指定する必要があります。

#### パラメーター:

##### **dest (入力)**

Destination オブジェクト。

##### **msg (入力)**

Message オブジェクト。

#### 戻り値:

Void

#### 例外:

- XMSEException
- MessageFormatException
- InvalidDestinationException

Send - 送信 (送達モード、優先順位、および存続時間を指定して、指定された宛先へ)

#### インターフェース:

```
void Send(IDestination dest,  
          IMessage msg,  
          DeliveryMode deliveryMode,  
          Int32 priority,  
          Int64 timeToLive) ;
```

メッセージ・プロデューサーが作成されたときに宛先が指定されていないメッセージ・プロデューサーを使用している場合は、指定された宛先にメッセージを送信します。指定された送達モード、優先順位、および存続時間を使用してメッセージを送信します。

通常、メッセージ・プロデューサーの作成時には宛先を指定しますが、宛先を指定しない場合、メッセージの送信ごとに宛先を指定する必要があります。

#### パラメーター:

##### **dest (入力)**

Destination オブジェクト。

##### **msg (入力)**

Message オブジェクト。

##### **deliveryMode (入力)**

メッセージの送達モード。以下の値のいずれかでなければなりません。

`DeliveryMode.Persistent`

`DeliveryMode.NonPersistent`

ブローカーへのリアルタイム接続の場合、値は `DeliveryMode.NonPersistent` である必要があります。

##### **priority (入力)**

メッセージの優先順位。値は 0 (最低優先順位) から 9 (最高優先順位) までの整数です。ブローカーへのリアルタイム接続では、値は無視されます。

##### **timeToLive (入力)**

メッセージの存続時間 (ミリ秒単位)。値 0 は、メッセージの有効期限がないことを意味します。ブローカーへのリアルタイム接続の場合、値は 0 である必要があります。

#### 戻り値:

Void

#### 例外:

- `XMSEException`
- `MessageFormatException`
- `InvalidDestinationException`
- `IllegalStateException`

## 継承されたプロパティおよびメソッド

以下のメソッドは、`IPropertyContext` インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IObjectMessage

オブジェクト・メッセージとは、本体が、直列化された Java または .NET オブジェクトからなるメッセージです。

#### 継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.IObjectMessage
```

## 関連資料

[オブジェクト・メッセージ](#)

オブジェクト・メッセージの本文には、シリアライズされた Java オブジェクトまたは .NET オブジェクトが含まれています。

## ドット・ネット プロパティ

ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<a href="#">OBJECT</a>	オブジェクト・メッセージの本体を形成するオブジェクトを取得および設定します。

*Object* - オブジェクトをバイトとして取得および設定

インターフェース:

```
System.Object Object
{
    get;
    set;
}

Byte[] GetObject();
```

オブジェクト・メッセージの本体を形成するオブジェクトを取得および設定します。

例外:

- [XMSEException](#)
- [MessageNotReadableException](#)
- [MessageEOFException](#)
- [MessageNotWritableException](#)

## 継承されたプロパティおよびメソッド

以下のプロパティは、[IMessage](#) インターフェースから継承されています。

[JMSCorrelationID](#)、[JMSDeliveryMode](#)、[JMSDestination](#)、[JMSExpiration](#)、[JMSMessageID](#)、[JMSPriority](#)、[JMSRedelivered](#)、[JMSReplyTo](#)、[JMSTimestamp](#)、[JMSType](#)、[Properties](#)

以下のメソッドは、[IMessage](#) インターフェースから継承されています。

[clearBody](#)、[clearProperties](#)、[PropertyExists](#)

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IPropertyContext

[IPropertyContext](#) は、プロパティを取得および設定するメソッドを含む抽象スーパークラスです。これらのメソッドは、その他のクラスによって継承されます。

継承の階層:

なし

## 方法

### メソッドの要約:

メソッド	説明
<a href="#">GetBooleanProperty</a>	指定した名前を持つ boolean プロパティの値を取得します。
<a href="#">GetByteProperty</a>	名前で識別されるバイト・プロパティの値を取得します。
<a href="#">GetBytesProperty</a>	名前で識別されるバイト配列プロパティの値を取得します。
<a href="#">GetCharProperty</a>	名前で識別される 2 バイト文字プロパティの値を取得します。
<a href="#">GetDoubleProperty</a>	名前で識別される倍精度浮動小数点プロパティの値を取得します。
<a href="#">GetFloatProperty</a>	名前で識別される浮動小数点プロパティの値を取得します。
<a href="#">GetIntProperty</a>	名前で識別される整数プロパティの値を取得します。
<a href="#">GetLongProperty</a>	名前で識別される長整数プロパティの値を取得します。
<a href="#">GetObjectProperty</a>	名前で識別されるプロパティの値およびデータ・タイプを取得します。
<a href="#">GetShortProperty</a>	名前で識別される短整数プロパティの値を取得します。
<a href="#">GetStringProperty</a>	名前で識別されるストリング・プロパティの値を取得します。
<a href="#">SetBooleanProperty</a>	名前で識別される boolean プロパティの値を設定します。
<a href="#">SetByteProperty</a>	名前で識別されるバイト・プロパティの値を設定します。
<a href="#">SetBytesProperty</a>	名前で識別されるバイト配列プロパティの値を設定します。
<a href="#">SetCharProperty</a>	名前で識別される 2 バイト文字プロパティの値を設定します。
<a href="#">SetDoubleProperty</a>	名前で識別される倍精度浮動小数点プロパティの値を設定します。
<a href="#">SetFloatProperty</a>	名前で識別される浮動小数点プロパティの値を設定取得します。
<a href="#">SetIntProperty</a>	名前で識別される整数プロパティの値を設定します。
<a href="#">SetLongProperty</a>	名前で識別される長整数プロパティの値を設定します。
<a href="#">SetObjectProperty</a>	名前で識別されるプロパティの値およびデータ・タイプを設定します。
<a href="#">SetShortProperty</a>	名前で識別される短整数プロパティの値を設定します。
<a href="#">SetStringProperty</a>	名前で識別されるストリング・プロパティの値を設定します。

*GetBooleanProperty* - boolean プロパティの取得

### インターフェース:

```
Boolean GetBooleanProperty(String property_name);
```

指定した名前を持つ boolean プロパティの値を取得します。

### パラメーター:

#### **property\_name** (入力)

プロパティの名前をカプセル化している String オブジェクト。

### 戻り値:

プロパティの値。

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

## GetByteProperty - バイト・プロパティの取得

### インターフェース:

```
Byte    GetByteProperty(String property_name) ;  
Int16   GetSignedByteProperty(String property_name) ;
```

名前で識別されるバイト・プロパティの値を取得します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

### 戻り値:

プロパティの値。

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

## GetBytesProperty - バイト配列プロパティの取得

### インターフェース:

```
Byte[]  GetBytesProperty(String property_name) ;
```

名前で識別されるバイト配列プロパティの値を取得します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

### 戻り値:

配列のバイト数。

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

## GetCharProperty - 文字プロパティの取得

### インターフェース:

```
Char    GetCharProperty(String property_name) ;
```

名前で識別される 2 バイト文字プロパティの値を取得します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

### 戻り値:

プロパティの値。

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

*GetDoubleProperty* - 倍精度浮動小数点プロパティーの取得

インターフェース:

```
Double GetDoubleProperty(String property_name) ;
```

名前で識別される倍精度浮動小数点プロパティーの値を取得します。

パラメーター:

**property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

戻り値:

プロパティーの値。

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException

*GetFloatProperty* - 浮動小数点プロパティーの取得

インターフェース:

```
Single GetFloatProperty(String property_name) ;
```

名前で識別される浮動小数点プロパティーの値を取得します。

パラメーター:

**property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

戻り値:

プロパティーの値。

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException

*GetIntProperty* - 整数プロパティーの取得

インターフェース:

```
Int32 GetIntProperty(String property_name) ;
```

名前で識別される整数プロパティーの値を取得します。

パラメーター:

**property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

戻り値:

プロパティーの値。

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException

## *GetLongProperty* - 長整数プロパティーの取得

### インターフェース:

```
Int64 GetLongProperty(String property_name) ;
```

名前で識別される長整数プロパティーの値を取得します。

### パラメーター:

#### **property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

### 戻り値:

プロパティーの値。

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

## *GetObjectProperty* - オブジェクト・プロパティーの取得

### インターフェース:

```
Object GetObjectProperty( String property_name) ;
```

名前で識別されるプロパティーの値およびデータ・タイプを取得します。

### パラメーター:

#### **property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

### 戻り値:

オブジェクト・タイプが以下のいずれかであるプロパティーの値。

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException

## *GetShortProperty* - 短整数プロパティーの取得

### インターフェース:

```
Int16 GetShortProperty(String property_name) ;
```

名前で識別される短整数プロパティーの値を取得します。

**パラメーター:**

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**戻り値:**

プロパティの値。

**スレッド・コンテキスト:**

サブクラスによって決定

**例外:**

- XMSEException

*GetStringProperty* - ストリング・プロパティの取得

**インターフェース:**

```
String GetStringValue(String property_name) ;
```

名前で識別されるストリング・プロパティの値を取得します。

**パラメーター:**

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**戻り値:**

プロパティの値を表すストリングをカプセル化している String オブジェクト。データ変換が必要な場合、この値は、変換後のストリングになります。

**スレッド・コンテキスト:**

サブクラスによって決定

**例外:**

- XMSEException

*SetBooleanProperty* - *boolean* プロパティの設定

**インターフェース:**

```
void SetBooleanProperty( String property_name, Boolean value) ;
```

名前で識別される boolean プロパティの値を設定します。

**パラメーター:**

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**value (入力)**

プロパティの値。

**戻り値:**

Void

**スレッド・コンテキスト:**

サブクラスによって決定

**例外:**

- XMSEException
- MessageNotWritableException

## SetByteProperty - バイト・プロパティの設定

### インターフェース:

```
void SetByteProperty( String property_name, Byte value) ;  
void SetSignedByteProperty( String property_name, Int16 value) ;
```

名前で識別されるバイト・プロパティの値を設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

#### **value (入力)**

プロパティの値。

### 戻り値:

Void

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException
- MessageNotWritableException

## SetBytesProperty - バイト配列プロパティの設定

### インターフェース:

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

名前で識別されるバイト配列プロパティの値を設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

#### **value (入力)**

バイトの配列であるプロパティの値。

### 戻り値:

Void

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException
- MessageNotWritableException

## SetCharProperty - 文字プロパティの設定

### インターフェース:

```
void SetCharProperty( String property_name, Char value) ;
```

名前で識別される 2 バイト文字プロパティの値を設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

#### **value (入力)**

プロパティの値。

戻り値:

Void

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException
- MessageNotWritableException

*SetDoubleProperty* - 倍精度浮動小数点プロパティの設定

インターフェース:

```
void SetDoubleProperty( String property_name, Double value) ;
```

名前で識別される倍精度浮動小数点プロパティの値を設定します。

パラメーター:

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**value (入力)**

プロパティの値。

戻り値:

Void

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException
- MessageNotWritableException

*SetFloatProperty* - 浮動小数点プロパティの設定

インターフェース:

```
void SetFloatProperty( String property_name, Single value) ;
```

名前で識別される浮動小数点プロパティの値を設定取得します。

パラメーター:

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**value (入力)**

プロパティの値。

戻り値:

Void

スレッド・コンテキスト:

サブクラスによって決定

例外:

- XMSEException
- MessageNotWritableException

## SetIntProperty - 整数プロパティーの設定

### インターフェース:

```
void SetIntProperty( String property_name, Int32 value) ;
```

名前で識別される整数プロパティーの値を設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

#### **value (入力)**

プロパティーの値。

### 戻り値:

Void

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException
- MessageNotWritableException

## SetLongProperty - 長整数プロパティーの設定

### インターフェース:

```
void SetLongProperty( String property_name, Int64 value) ;
```

名前で識別される長整数プロパティーの値を設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

#### **value (入力)**

プロパティーの値。

### 戻り値:

Void

### スレッド・コンテキスト:

サブクラスによって決定

### 例外:

- XMSEException
- MessageNotWritableException

## SetObjectProperty - オブジェクト・プロパティーの設定

### インターフェース:

```
void SetObjectProperty( String property_name, Object value) ;
```

名前で識別されるプロパティーの値およびデータ・タイプを設定します。

### パラメーター:

#### **property\_name (入力)**

プロパティーの名前をカプセル化している String オブジェクト。

#### **objectType (入力)**

プロパティーの値。以下のオブジェクト・タイプのいずれかでなければなりません。

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**value (入力)**

バイトの配列としてのプロパティの値。

**length (入力)**

配列のバイト数。

**戻り値:**

Void

**スレッド・コンテキスト:**

サブクラスによって決定

**例外:**

- XMSEException
- MessageNotWritableException

*SetShortProperty* - 短整数プロパティの設定

**インターフェース:**

```
void SetShortProperty( String property_name, Int16 value) ;
```

名前で識別される短整数プロパティの値を設定します。

**パラメーター:**

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

**value (入力)**

プロパティの値。

**戻り値:**

Void

**スレッド・コンテキスト:**

サブクラスによって決定

**例外:**

- XMSEException
- MessageNotWritableException

*SetStringProperty* - ストリング・プロパティの設定

**インターフェース:**

```
void SetStringProperty( String property_name, String value);
```

名前で識別されるストリング・プロパティの値を設定します。

**パラメーター:**

**property\_name (入力)**

プロパティの名前をカプセル化している String オブジェクト。

## value (入力)

プロパティの値を表す文字列をカプセル化している String オブジェクト。

## 戻り値:

Void

## スレッド・コンテキスト:

サブクラスによって決定

## 例外:

- XMSEException
- MessageNotWritableException

## IQueueBrowser

アプリケーションは、キュー・ブラウザーを使用して、キュー上のメッセージを参照します。その際にメッセージは除去されません。

## 継承の階層:

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
|
+----+ IBM.XMS.IQueueBrowser
```

## ドット・ネット プロパティ

### ドット・ネット プロパティの要約:

ドット・ネット プロパティ

[MessageSelector](#)

[Queue](#)

### 説明

キュー・ブラウザーのメッセージ・セレクターを取得します。

キュー・ブラウザーに関連付けられたキューを、キューを表す宛先オブジェクトとして取得します。

## MessageSelector - メッセージ・セレクターの取得

### インターフェース:

```
String MessageSelector
{
    get;
}
```

キュー・ブラウザーのメッセージ・セレクターを取得します。

メッセージ・セレクターは、メッセージ・セレクター式をカプセル化している String オブジェクトです。データ変換が必要な場合、この値は、変換後のメッセージ・セレクター式になります。キュー・ブラウザーにメッセージ・セレクターが存在しない場合、このメソッドはヌルの String オブジェクトを返します。

### 例外:

- XMSEException

## Queue - キューの取得

### インターフェース:

```
IDestination Queue
{
    get;
}
```

キュー・ブラウザーに関連付けられたキューを、キューを表す宛先オブジェクトとして取得します。

#### 例外:

- XMSEException

## 方法

#### メソッドの要約:

メソッド	説明
<a href="#">Close</a>	キュー・ブラウザーを閉じます。
<a href="#">GetEnumerator</a>	キュー上のメッセージのリストを取得します。

*Close* - キュー・ブラウザーのクローズ

#### インターフェース:

```
void Close();
```

キュー・ブラウザーを閉じます。

アプリケーションが、既に閉じているキュー・ブラウザーを閉じようとした場合、呼び出しは無視されます。

#### パラメーター:

なし

#### 戻り値:

Void

#### 例外:

- XMSEException

*GetEnumerator* - メッセージの取得

#### インターフェース:

```
IEnumerator GetEnumerator();
```

キュー上のメッセージのリストを取得します。

このメソッドは、*Message* オブジェクトのリストをカプセル化する列挙子を戻します。*Message* オブジェクトの順序は、メッセージがキューから取り出される順序と同じです。アプリケーションは列挙子を使用して、各メッセージを順番に参照できます。

メッセージがキューに書き込まれたり削除されたりすると、列挙子は動的に更新されます。アプリケーションがキュー上の次のメッセージを参照するために *IEnumerator.MoveNext()* を呼び出すたびに、メッセージはキューの現在の内容を反映しています。

アプリケーションがキュー・ブラウザーに対してこのメソッドを複数回呼び出すと、そのたびに新しい列挙子が返されます。したがって、アプリケーションは複数の列挙子を使用してキューのメッセージを参照し、キュー内の複数の位置を維持することができます。

#### パラメーター:

なし

#### 戻り値:

Iterator オブジェクト。

#### 例外:

- XMSEException

## 継承されたプロパティおよびメソッド

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## 要求者

アプリケーションはリクエスターを使用して、要求メッセージを送信し、応答を待機して受信します。

継承の階層:

なし

## コンストラクター

コンストラクターの要約:

コンストラクター	説明
<a href="#">Requestor</a>	リクエスターを作成します。

*Requestor* - リクエスターの作成

インターフェース:

```
Requestor(ISession sess, IDestination dest);
```

リクエスターを作成します。

パラメーター:

**sess (入力)**

Session オブジェクト。セッションはトランザクション化されてはならず、以下の肯定応答モードのいずれかでなければなりません。

`AcknowledgeMode.AutoAcknowledge`  
`AcknowledgeMode.DupsOkAcknowledge`

**dest (入力)**

アプリケーションが要求メッセージを送信できる宛先を表す Destination オブジェクト。

スレッド・コンテキスト:

リクエスターに関連付けられたセッション

例外:

- `XMSEException`

## 方法

メソッドの要約:

メソッド	説明
<a href="#">Close</a>	リクエスターを閉じます。
<a href="#">Request</a>	要求メッセージを送信してから、その要求メッセージを受信するアプリケーションからの応答を待機して応答を受信します。

*Close* - リクエスターのクローズ

インターフェース:

```
void Close();
```

リクエスターを閉じます。

アプリケーションが、既に閉じているリクエスターを閉じようとした場合、呼び出しは無視されます。

**注:** アプリケーションがリクエスターを閉じたとき、関連したセッションは閉じません。この点で、XMSの動作はJMSとは異なります。

**パラメーター:**

なし

**戻り値:**

Void

**スレッド・コンテキスト:**

任意

**例外:**

- XMSEException

*Request* - 応答の要求

**インターフェース:**

```
IMessage Request(IMessage requestMessage);
```

要求メッセージを送信してから、その要求メッセージを受信するアプリケーションからの応答を待機して応答を受信します。

このメソッドへの呼び出しは、応答が受信されるまで、またはセッションが終了するまでの、いずれか早い方の時点までブロックします。

**パラメーター:**

**requestMessage (入力)**

要求メッセージをカプセル化している Message オブジェクト。

**戻り値:**

応答メッセージをカプセル化している Message オブジェクトを指すポインター。

**スレッド・コンテキスト:**

リクエスターに関連付けられたセッション

**例外:**

- XMSEException

## ResourceAllocationException

継承の階層:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.ResourceAllocationException
```

XMSがこの例外をスローするのは、メソッドが必要とするリソースをXMSが割り振ることができない場合です。

## 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSEException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## SecurityException

継承の階層:

```
IBM.XMS.XMSEException
|
```

```
+----IBM.XMS.XMSException
      |
      +----IBM.XMS.SecurityException
```

XMS がこの例外をスローするのは、アプリケーションを認証するために指定したユーザー ID とパスワードがリジェクトされた場合です。XMS は、権限検査が不合格になり、そのためにメソッドを完了できない場合にもこの例外をスローします。

## 継承されたプロパティおよびメソッド

以下のメソッドは、[XMSException](#) インターフェースから継承されています。

[GetErrorCode](#)、[GetLinkedException](#)

## ISession

セッションとは、メッセージ送受信のための単一スレッドのコンテキストのことです。

継承の階層:

```
IBM.XMS.IPropertyContext
      |
      +----IBM.XMS.ISession
```

Session オブジェクトの XMS 定義プロパティのリストについては、[198 ページの『Session のプロパティ』](#)を参照してください。

## .NET プロパティ

.NET プロパティの要約:

.NET プロパティ	説明
<a href="#">AcknowledgeMode</a>	セッションの肯定応答モードを取得します。
<a href="#">Transacted</a>	セッションがトランザクション化されているかどうかを判別します。

*AcknowledgeMode* - 肯定応答モードの取得

インターフェース:

```
AcknowledgeMode AcknowledgeMode
{
    get;
}
```

セッションの肯定応答モードを取得します。

肯定応答モードは、セッション作成時に指定されます。

セッションがトランザクション化されていない場合、肯定応答モードは以下の値のいずれかです。

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

肯定応答モードについて詳しくは、[28 ページの『メッセージ肯定応答』](#)を参照してください。

トランザクション化されているセッションには、肯定応答モードはありません。セッションが処理された場合、メソッドは代わりに `AcknowledgeMode.SessionTransacted` を戻します。

例外:

- `XMSException`

Transacted - 処理済みであるかどうかの判別

インターフェース:

```
Boolean Transacted
{
    get;
}
```

セッションがトランザクション化されているかどうかを判別します。

トランザクション化されている状態は以下のとおりです。

- セッションがトランザクション化されている場合は、True です。
- セッションがトランザクション化されていない場合は、False です。

ブローカーへのリアルタイム接続の場合、常にメソッドは False を戻します。

例外:

- XMSEException

## 方法

メソッドの要約:

メソッド	説明
<a href="#">Close</a>	セッションを閉じます。
<a href="#">Commit</a>	現在のトランザクションで処理されたすべてのメッセージをコミットします。
<a href="#">CreateBrowser</a>	指定されたキューのキュー・ブラウザーを作成します。
<a href="#">CreateBrowser</a>	メッセージ・セレクターを使用して、指定されたキューのキュー・ブラウザーを作成します。
<a href="#">CreateBytesMessage</a>	バイト・メッセージを作成します。
<a href="#">CreateConsumer</a>	指定された宛先のメッセージ・コンシューマーを作成します。
<a href="#">CreateConsumer</a>	メッセージ・セレクターを使用して、指定された宛先のメッセージ・コンシューマーを作成します。
<a href="#">CreateConsumer</a>	メッセージ・セレクターを使用し、宛先がトピックの場合はメッセージ・コンシューマーが自身の接続により公開されたメッセージを受信するかどうかを指定して、指定された宛先のメッセージ・コンシューマーを作成します。
<a href="#">CreateDurableSubscriber</a>	指定されたトピックの永続サブスクライバーを作成します。
<a href="#">CreateDurableSubscriber</a>	メッセージ・セレクターを使用し、永続サブスクライバーが自身の接続により公開されたメッセージを受信するかどうかを指定して、指定されたトピックの永続サブスクライバーを作成します。
<a href="#">CreateMapMessage</a>	マップ・メッセージを作成します。
<a href="#">CreateMessage</a>	本体を持たないメッセージを作成します。
<a href="#">CreateObjectMessage</a>	オブジェクト・メッセージを作成します。
<a href="#">CreateProducer</a>	メッセージを指定された宛先へ送信するメッセージ・プロデューサーを作成します。
<a href="#">CreateQueue</a>	メッセージング・サーバー内のキューを表すための Destination オブジェクトを作成します。
<a href="#">CreateStreamMessage</a>	ストリーム・メッセージを作成します。

メソッド	説明
<a href="#">CreateTemporaryQueue</a>	一時キューを作成します。
<a href="#">CreateTemporaryTopic</a>	一時トピックを作成します。
<a href="#">CreateTextMessage</a>	本体が空であるテキスト・メッセージを作成します。
<a href="#">CreateTextMessage</a>	本体が指定されたテキストで初期化されているテキスト・メッセージを作成します。
<a href="#">CreateTopic</a>	トピックを表すための <i>Destination</i> オブジェクトを作成します。
<a href="#">Recover</a>	セッションを回復します。
<a href="#">Rollback</a>	現在のトランザクションで処理されたすべてのメッセージをロールバックします。
<a href="#">Unsubscribe</a>	永続サブスクリプションを削除します。

*Close* - セッションのクローズ

インターフェース:

```
void Close();
```

セッションを閉じます。セッションがトランザクション化されている場合、進行中のトランザクションはロールバックされます。

アプリケーションが、既に閉じているセッションを閉じようとした場合、呼び出しは無視されます。

パラメーター:

なし

戻り値:

Void

スレッド・コンテキスト:

任意

例外:

- *XMSEException*

*Commit* - コミット

インターフェース:

```
void Commit();
```

現在のトランザクションで処理されたすべてのメッセージをコミットします。

セッションは、トランザクション化セッションでなければなりません。

パラメーター:

なし

戻り値:

Void

例外:

- *XMSEException*
- *IllegalStateException*
- *TransactionRolledBackException*

## CreateBrowser - キュー・ブラウザの作成

### インターフェース:

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

指定されたキューのキュー・ブラウザを作成します。

### パラメーター:

#### queue (入力)

キューを表す Destination オブジェクト。

### 戻り値:

QueueBrowser オブジェクト。

### 例外:

- XMSEException
- InvalidDestinationException

## CreateBrowser - キュー・ブラウザの作成 (メッセージ・セレクターを使用)

### インターフェース:

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

メッセージ・セレクターを使用して、指定されたキューのキュー・ブラウザを作成します。

### パラメーター:

#### queue (入力)

キューを表す Destination オブジェクト。

#### selector (入力)

メッセージ・セレクター式をカプセル化している String オブジェクト。メッセージ・セレクター式に一致するプロパティを持つメッセージのみが、キュー・ブラウザに配信されます。

String オブジェクトがヌルであるとは、キュー・ブラウザ用のメッセージ・セレクターが存在しないという意味です。

### 戻り値:

QueueBrowser オブジェクト。

### 例外:

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

## CreateBytesMessage - バイト・メッセージの作成

### インターフェース:

```
IBytesMessage CreateBytesMessage();
```

バイト・メッセージを作成します。

### パラメーター:

なし

### 戻り値:

BytesMessage オブジェクト。

### 例外:

- XMSEException
- IllegalStateException (セッションは終了しています)

## CreateConsumer - コンシューマーの作成

### インターフェース:

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

指定された宛先のメッセージ・コンシューマーを作成します。

### パラメーター:

#### dest (入力)

Destination オブジェクト。

### 戻り値:

MessageConsumer オブジェクト。

### 例外:

- XMSException
- InvalidDestinationException

## CreateConsumer - コンシューマーの作成 (メッセージ・セレクターを使用)

### インターフェース:

```
IMessageConsumer CreateConsumer(IDestination dest,  
                                String selector) ;
```

メッセージ・セレクターを使用して、指定された宛先のメッセージ・コンシューマーを作成します。

### パラメーター:

#### dest (入力)

Destination オブジェクト。

#### selector (入力)

メッセージ・セレクター式をカプセル化している String オブジェクト。メッセージ・セレクター式に一致するプロパティを持つメッセージのみが、メッセージ・コンシューマーに配信されます。

String オブジェクトがヌルであるとは、メッセージ・コンシューマー用のメッセージ・セレクターが存在しないという意味です。

### 戻り値:

MessageConsumer オブジェクト。

### 例外:

- XMSException
- InvalidDestinationException
- InvalidSelectorException

## CreateConsumer - コンシューマーの作成 (メッセージ・セレクターおよびローカル・メッセージ・フラグを使用)

### インターフェース:

```
IMessageConsumer CreateConsumer(IDestination dest,  
                                String selector,  
                                Boolean noLocal) ;
```

メッセージ・セレクターを使用し、宛先がトピックの場合はメッセージ・コンシューマーが自身の接続により公開されたメッセージを受信するかどうかを指定して、指定された宛先のメッセージ・コンシューマーを作成します。

### パラメーター:

#### dest (入力)

Destination オブジェクト。

### **selector (入力)**

メッセージ・セレクター式をカプセル化している String オブジェクト。メッセージ・セレクター式に一致するプロパティを持つメッセージのみが、メッセージ・コンシューマーに配信されます。

String オブジェクトがヌルであるとは、メッセージ・コンシューマー用のメッセージ・セレクターが存在しないという意味です。

### **noLocal (入力)**

値 True は、メッセージ・コンシューマーが、自身の接続により公開されたメッセージを受信しないことを意味します。値 False は、メッセージ・コンシューマーが、自身の接続により公開されたメッセージを受信することを意味します。デフォルト値は False です。

### **戻り値:**

MessageConsumer オブジェクト。

### **例外:**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

## *CreateDurableSubscriber* - 永続サブスクライバーの作成

### **インターフェース:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                          String subscription) ;
```

指定されたトピックの永続サブスクライバーを作成します。

ブローカーへのリアルタイム接続の場合、このメソッドは無効です。

永続サブスクライバーについて詳しくは、[36 ページの『永続サブスクライバー』](#)を参照してください。

### **パラメーター:**

#### **dest (入力)**

トピックを表す Destination オブジェクト。トピックは、一時トピックであってはなりません。

#### **subscription (入力)**

永続サブスクリプションを識別する名前をカプセル化している String オブジェクト。名前は、接続のクライアント ID 内で固有である必要があります。

### **戻り値:**

永続サブスクライバーを表す MessageConsumer オブジェクト。

### **例外:**

- XMSEException
- InvalidDestinationException

## *CreateDurableSubscriber* - 永続サブスクライバーの作成 (メッセージ・セレクターおよびローカル・メッセージ・フラグを使用)

### **インターフェース:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                          String subscription,  
                                          String selector,  
                                          Boolean noLocal) ;
```

メッセージ・セレクターを使用し、永続サブスクライバーが自身の接続により公開されたメッセージを受信するかどうかを指定して、指定されたトピックの永続サブスクライバーを作成します。

ブローカーへのリアルタイム接続の場合、このメソッドは無効です。

永続サブスクライバーについて詳しくは、[36 ページの『永続サブスクライバー』](#)を参照してください。

#### パラメーター:

##### **dest (入力)**

トピックを表す Destination オブジェクト。トピックは、一時トピックであってはなりません。

##### **subscription (入力)**

永続サブスクリプションを識別する名前をカプセル化している String オブジェクト。名前は、接続のクライアント ID 内で固有である必要があります。

##### **selector (入力)**

メッセージ・セレクター式をカプセル化している String オブジェクト。メッセージ・セレクター式に一致するプロパティを持つメッセージのみが、永続サブスクライバーに配信されます。

String オブジェクトがヌルであるとは、永続サブスクライバー用のメッセージ・セレクターが存在しないという意味です。

##### **noLocal (入力)**

値 True は、永続サブスクライバーが、自身の接続により公開されたメッセージを受信しないことを意味します。値 False は、永続サブスクライバーが、自身の接続により公開されたメッセージを受信することを意味します。デフォルト値は False です。

#### 戻り値:

永続サブスクライバーを表す MessageConsumer オブジェクト。

#### 例外:

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

#### CreateMapMessage - マップ・メッセージの作成

##### インターフェース:

```
IMapMessage CreateMapMessage();
```

マップ・メッセージを作成します。

##### パラメーター:

なし

##### 戻り値:

MapMessage オブジェクト。

#### 例外:

- XMSEException
- IllegalStateException (セッションは終了しています)

#### CreateMessage - メッセージの作成

##### インターフェース:

```
IMessage CreateMessage();
```

本体を持たないメッセージを作成します。

##### パラメーター:

なし

##### 戻り値:

Message オブジェクト。

#### 例外:

- XMSEException
- IllegalStateException (セッションは終了しています)

## CreateObjectMessage - オブジェクト・メッセージの作成

### インターフェース:

```
IObjectMessage CreateObjectMessage();
```

オブジェクト・メッセージを作成します。

### パラメーター:

なし

### 戻り値:

ObjectMessage オブジェクト。

### 例外:

- XMSEException
- IllegalStateException (セッションは終了しています)

## CreateProducer - プロデューサーの作成

### インターフェース:

```
IMessageProducer CreateProducer(IDestination dest) ;
```

メッセージを指定された宛先へ送信するメッセージ・プロデューサーを作成します。

### パラメーター:

#### dest (入力)

Destination オブジェクト。

ヌルの Destination オブジェクトを指定すると、宛先のないメッセージ・プロデューサーが作成されます。この場合アプリケーションは、メッセージを送信するためにメッセージ・プロデューサーを使用するたびに、宛先を指定する必要があります。

### 戻り値:

MessageProducer オブジェクト。

### 例外:

- XMSEException
- InvalidDestinationException

## CreateQueue - キューの作成

### インターフェース:

```
IDestination CreateQueue(String queue) ;
```

メッセージング・サーバー内のキューを表すための Destination オブジェクトを作成します。

このメソッドは、メッセージング・サーバー内にキューを作成しません。アプリケーションがこのメソッドを呼び出すためには、その前にキューを作成する必要があります。

### パラメーター:

#### queue (入力)

キューの名前をカプセル化している String オブジェクト、またはキューを識別する Uniform Resource Identifier (URI) をカプセル化している String オブジェクト。

### 戻り値:

キューを表す Destination オブジェクト。

### 例外:

- XMSEException

## CreateStreamMessage - ストリーム・メッセージの作成

### インターフェース:

```
IStreamMessage CreateStreamMessage();
```

ストリーム・メッセージを作成します。

### パラメーター:

なし

### 戻り値:

StreamMessage オブジェクト。

### 例外:

- XMSEException
- XMS\_ILLEGAL\_STATE\_EXCEPTION

## CreateTemporaryQueue - 一時キューの作成

### インターフェース:

```
IDestination CreateTemporaryQueue();
```

一時キューを作成します。

一時キューの範囲は接続です。接続によって作成されたセッションのみが、一時キューを使用できます。

一時キューは、明示的に削除されるまで、あるいは接続が終了するまで存続します。

一時キューについては詳しくは、[34 ページの『一時宛先』](#)を参照してください。

### パラメーター:

なし

### 戻り値:

一時キューを表す Destination オブジェクト。

### 例外:

- XMSEException

## CreateTemporaryTopic - 一時トピックの作成

### インターフェース:

```
IDestination CreateTemporaryTopic();
```

一時トピックを作成します。

一時トピックの範囲は接続です。接続によって作成されたセッションのみが、一時トピックを使用できます。

一時トピックは、明示的な削除または接続終了のいずれかが発生するまで存在します。

一時トピックについては詳しくは、[34 ページの『一時宛先』](#)を参照してください。

### パラメーター:

なし

### 戻り値:

一時トピックを表す Destination オブジェクト。

### 例外:

- XMSEException

## CreateTextMessage - テキスト・メッセージの作成

### インターフェース:

```
ITextMessage CreateTextMessage();
```

本体が空であるテキスト・メッセージを作成します。

### パラメーター:

なし

### 戻り値:

TextMessage オブジェクト。

### 例外:

- XMSEException

## CreateTextMessage - テキスト・メッセージの作成 (初期化済み)

### インターフェース:

```
ITextMessage CreateTextMessage(String initialValue);
```

本体が指定されたテキストで初期化されているテキスト・メッセージを作成します。

### パラメーター:

#### **initialValue (入力)**

テキスト・メッセージの本体を初期化するためのテキストをカプセル化している String オブジェクト。

なし

### 戻り値:

TextMessage オブジェクト。

### 例外:

- XMSEException

## CreateTopic - トピックの作成

### インターフェース:

```
IDestination CreateTopic(String topic) ;
```

トピックを表すための Destination オブジェクトを作成します。

### パラメーター:

#### **topic (入力)**

トピックの名前をカプセル化している String オブジェクト、またはトピックを識別する Uniform Resource Identifier (URI) をカプセル化している String オブジェクト。

### 戻り値:

トピックを表す Destination オブジェクト。

### 例外:

- XMSEException

## Recover - 回復

### インターフェース:

```
void Recover();
```

セッションを回復します。メッセージ配信が一旦停止され、その後応答されていない最も古いメッセージを使用して再開されます。

セッションは、トランザクション化セッションであってはなりません。

セッションの回復について詳しくは、[28 ページの『メッセージ肯定応答』](#)を参照してください。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException
- IllegalStateException

*Rollback* - ロールバック

**インターフェース:**

```
void Rollback();
```

現在のトランザクションで処理されたすべてのメッセージをロールバックします。

セッションは、トランザクション化セッションでなければなりません。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException
- IllegalStateException

*Unsubscribe* - アンサブスクライブ

**インターフェース:**

```
void Unsubscribe(String subscription);
```

永続サブスクリプションを削除します。メッセージング・サーバーは保守している永続サブスクリプションのレコードを削除し、永続サブスクライバーにこれ以降メッセージを送信しなくなります。

アプリケーションは、以下のいずれの状況でも、永続サブスクリプションを削除することはできません。

- 永続サブスクリプションのアクティブなメッセージ・コンシューマーがあるとき
- コンシュームされたメッセージが保留中のトランザクションの一部であるとき
- コンシュームされたメッセージの確認応答がなかったとき

ブローカーへのリアルタイム接続の場合、このメソッドは無効です。

**パラメーター:**

**subscription (入力)**

永続サブスクリプションを識別する名前をカプセル化している String オブジェクト。

**戻り値:**

Void

**例外:**

- XMSEException
- InvalidDestinationException
- IllegalStateException

## 継承されたプロパティおよびメソッド

以下のメソッドは、`IPROPERTYCONTEXT` インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## IStreamMessage

ストリーム・メッセージとは、本体が値のストリームで構成されるメッセージです。それぞれの値に、関連付けられたデータ・タイプがあります。本体の内容は、順番に読み書きされます。

継承の階層:



アプリケーションがメッセージ・ストリームから値を読み取る場合、その値は XMS によって別のデータ・タイプに変換されることがあります。この形式の暗黙の型変換については、[81 ページ](#)の『ストリーム・メッセージ』を参照してください。

### 関連資料

[ストリーム・メッセージ](#)

ストリーム・メッセージの本文には、値のストリームが含まれており、それぞれの値には関連付けられたデータ型があります。

## 方法

メソッドの要約:

メソッド	説明
<a href="#">ReadBoolean</a>	メッセージ・ストリームからブール値を読み取ります。
<a href="#">ReadByte</a>	メッセージ・ストリームから符号付き 8 ビット整数を読み取ります。
<a href="#">ReadBytes</a>	メッセージ・ストリームからバイトの配列を読み取ります。
<a href="#">ReadChar</a>	メッセージ・ストリームから 2 バイト文字を読み取ります。
<a href="#">ReadDouble</a>	メッセージ・ストリームから 8 バイトの倍精度浮動小数点数を読み取ります。
<a href="#">ReadFloat</a>	メッセージ・ストリームから 4 バイトの浮動小数点数を読み取ります。
<a href="#">ReadInt</a>	メッセージ・ストリームから符号付き 32 ビット整数を読み取ります。
<a href="#">ReadLong</a>	メッセージ・ストリームから符号付き 64 ビット整数を読み取ります。
<a href="#">ReadObject</a>	メッセージ・ストリームから値を読み取り、そのデータ・タイプを戻します。
<a href="#">ReadShort</a>	メッセージ・ストリームから符号付き 16 ビット整数を読み取ります。
<a href="#">ReadString</a>	メッセージ・ストリームからストリングを読み取ります。
<a href="#">Reset</a>	メッセージの本体を読み取り専用モードにして、カーソルをメッセージ・ストリームの先頭に位置変更します。
<a href="#">WriteBoolean</a>	メッセージ・ストリームへブール値を書き込みます。
<a href="#">WriteByte</a>	メッセージ・ストリームへバイトを書き込みます。

メソッド	説明
<a href="#">WriteBytes</a>	メッセージ・ストリームへバイトの配列を書き込みます。
<a href="#">WriteChar</a>	文字を、上位バイトを先にして、2 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteDouble</a>	倍精度浮動小数点数を長整数に変換し、その長整数を、上位バイトを先にして、8 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteFloat</a>	浮動小数点数を整数に変換し、その整数を、上位バイトを先にして、4 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteInt</a>	整数を、上位バイトを先にして、4 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteLong</a>	長整数を、上位バイトを先にして、8 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteObject</a>	メッセージ・ストリームに、指定されたデータ・タイプの値を書き込みます。
<a href="#">WriteShort</a>	短整数を、上位バイトを先にして、2 バイトでメッセージ・ストリームに書き込みます。
<a href="#">WriteString</a>	メッセージ・ストリームへストリングを書き込みます。

*ReadBoolean* - ブール値の読み取り

インターフェース:

```
Boolean ReadBoolean();
```

メッセージ・ストリームからブール値を読み取ります。

パラメーター:

なし

戻り値:

読み取られるブール値。

例外:

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - バイトの読み取り

インターフェース:

```
Int16 ReadSignedByte();
Byte ReadByte();
```

メッセージ・ストリームから符号付き 8 ビット整数を読み取ります。

パラメーター:

なし

戻り値:

読み取られるバイト。

例外:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## ReadBytes - バイトの読み取り

### インターフェース:

```
Int32 ReadBytes(Byte[] array);
```

メッセージ・ストリームからバイトの配列を読み取ります。

### パラメーター:

#### array (入力)

読み取られるバイトの配列が収容されているバッファーとそのバッファーの長さ (単位: バイト)。

配列のバイト数が、バッファーの長さ以下である場合は、配列全体がバッファーに読み取られます。配列のバイト数がバッファーの長さを超える場合、配列の一部でバッファーはいっぱいになり、内部カーソルは次に読み取られるバイトの位置をマークします。次に readBytes() を呼び出すと、カーソルの現在位置から始まる配列から、バイトが読み取られます。

入力に NULL ポインターを指定すると、呼び出しではそのバイトの配列は読み取られずにスキップオーバーされます。

### 戻り値:

バッファーに読み取るバイト数。バッファーが部分的に埋まっている場合は、値はバッファーの長さよりも小さく、配列内に読み取るバイトが残っていないことを示します。呼び出し前の配列に読み取り可能なバイトが残っていない場合、値は XMSC\_END\_OF\_BYTEARRAY です。

入力に NULL ポインターを指定すると、メソッドは値を戻しません。

### 例外:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## ReadChar - 文字の読み取り

### インターフェース:

```
Char ReadChar();
```

メッセージ・ストリームから 2 バイト文字を読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる文字。

### 例外:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## ReadDouble - 倍精度浮動小数点数の読み取り

### インターフェース:

```
Double ReadDouble();
```

メッセージ・ストリームから 8 バイトの倍精度浮動小数点数を読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる倍精度浮動小数点数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadFloat* - 浮動小数点数の読み取り

**インターフェース:**

```
Single ReadFloat();
```

メッセージ・ストリームから 4 バイトの浮動小数点数を読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる浮動小数点数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 整数の読み取り

**インターフェース:**

```
Int32 ReadInt();
```

メッセージ・ストリームから符号付き 32 ビット整数を読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる整数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadLong* - 長整数の読み取り

**インターフェース:**

```
Int64 ReadLong();
```

メッセージ・ストリームから符号付き 64 ビット整数を読み取ります。

**パラメーター:**

なし

**戻り値:**

読み取られる長整数。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

## *ReadObject* - オブジェクトの読み取り

### インターフェース:

```
Object ReadObject();
```

メッセージ・ストリームから値を読み取り、そのデータ・タイプを戻します。

### パラメーター:

なし

### 戻り値:

値。以下のオブジェクト・タイプのいずれかです。

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

### 例外:

XMSEException

## *ReadShort* - 短整数の読み取り

### インターフェース:

```
Int16 ReadShort();
```

メッセージ・ストリームから符号付き 16 ビット整数を読み取ります。

### パラメーター:

なし

### 戻り値:

読み取られる短整数。

### 例外:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## *ReadString* - スtringの読み取り

### インターフェース:

```
String ReadString();
```

メッセージ・ストリームからStringを読み取ります。必要な場合、XMS がString内の文字をローカル・コード・ページに変換します。

### パラメーター:

なし

### 戻り値:

読み取られるStringをカプセル化しているStringオブジェクト。データ変換が必要な場合、これは変換後のStringです。

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*Reset* - リセット

**インターフェース:**

```
void Reset();
```

メッセージの本体を読み取り専用モードにして、カーソルをメッセージ・ストリームの先頭に位置変更します。

**パラメーター:**

なし

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*WriteBoolean* - ブール値の書き込み

**インターフェース:**

```
void WriteBoolean(Boolean value);
```

メッセージ・ストリームへブール値を書き込みます。

**パラメーター:****value (入力)**

書き込まれるブール値。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteByte* - バイトの書き込み

**インターフェース:**

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

メッセージ・ストリームへバイトを書き込みます。

**パラメーター:****value (入力)**

書き込まれるバイト。

**戻り値:**

Void

**例外:**

- XMSEException

- MessageNotWritableException

*WriteBytes* - 複数バイトの書き込み

インターフェース:

```
void WriteBytes(Byte[] value);
```

メッセージ・ストリームへバイトの配列を書き込みます。

パラメーター:

**value (入力)**

書き込まれるバイト配列。

**length (入力)**

配列のバイト数。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

*WriteChar* - 文字の書き込み

インターフェース:

```
void WriteChar(Char value);
```

文字を、上位バイトを先にして、2バイトでメッセージ・ストリームに書き込みます。

パラメーター:

**value (入力)**

書き込まれる文字。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

*WriteDouble* - 倍精度浮動小数点数の書き込み

インターフェース:

```
void WriteDouble(Double value);
```

倍精度浮動小数点数を長整数に変換し、その長整数を、上位バイトを先にして、8バイトでメッセージ・ストリームに書き込みます。

パラメーター:

**value (入力)**

書き込まれる倍精度浮動小数点数。

戻り値:

Void

例外:

- XMSEException
- MessageNotWritableException

## WriteFloat - 浮動小数点数の書き込み

### インターフェース:

```
void WriteFloat(Single value);
```

浮動小数点数を整数に変換し、その整数を、上位バイトを先にして、4 バイトでメッセージ・ストリームに書き込みます。

### パラメーター:

#### value (入力)

書き込まれる浮動小数点数。

### 戻り値:

Void

### 例外:

- XMSEException
- MessageNotWritableException

## WriteInt - 整数の書き込み

### インターフェース:

```
void WriteInt(Int32 value);
```

整数を、上位バイトを先にして、4 バイトでメッセージ・ストリームに書き込みます。

### パラメーター:

#### value (入力)

書き込まれる整数。

### 戻り値:

Void

### 例外:

- XMSEException
- MessageNotWritableException

## WriteLong - 長整数の書き込み

### インターフェース:

```
void WriteLong(Int64 value);
```

長整数を、上位バイトを先にして、8 バイトでメッセージ・ストリームに書き込みます。

### パラメーター:

#### value (入力)

書き込まれる長整数。

### 戻り値:

Void

### 例外:

- XMSEException
- MessageNotWritableException

## WriteObject - オブジェクトの書き込み

### インターフェース:

```
void WriteObject(Object value);
```

メッセージ・ストリームに、指定されたデータ・タイプの値を書き込みます。

**パラメーター:**

**objectType (入力)**

値。以下のオブジェクト・タイプのいずれかでなければなりません。

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**value (入力)**

書き込まれる値を含むバイトの配列。

**length (入力)**

配列のバイト数。

**戻り値:**

Void

**例外:**

- XMSEException

*WriteShort* - 短整数の書き込み

**インターフェース:**

```
void WriteShort(Int16 value);
```

短整数を、上位バイトを先にして、2 バイトでメッセージ・ストリームに書き込みます。

**パラメーター:**

**value (入力)**

書き込まれる短整数。

**戻り値:**

Void

**例外:**

- XMSEException
- MessageNotWritableException

*WriteString* - スtringの書き込み

**インターフェース:**

```
void WriteString(String value);
```

メッセージ・ストリームへStringを書き込みます。

**パラメーター:**

**value (入力)**

書き込まれるStringをカプセル化しているString オブジェクト。

**戻り値:**

Void

#### 例外:

- XMSException
- MessageNotWritableException

### 継承されたプロパティおよびメソッド

以下のプロパティは、[IMessage](#) インターフェースから継承されています。

[JMSCorrelationID](#)、[JMSDeliveryMode](#)、[JMSDestination](#)、[JMSExpiration](#)、[JMSMessageID](#)、[JMSPriority](#)、[JMSRedelivered](#)、[JMSReplyTo](#)、[JMSTimestamp](#)、[JMSType](#)、[Properties](#)

以下のメソッドは、[IMessage](#) インターフェースから継承されています。

[clearBody](#)、[clearProperties](#)、[PropertyExists](#)

以下のメソッドは、[IPropertyContext](#) インターフェースから継承されています。

[GetBooleanProperty](#)、[GetByteProperty](#)、[GetBytesProperty](#)、[GetCharProperty](#)、[GetDoubleProperty](#)、[GetFloatProperty](#)、[GetIntProperty](#)、[GetLongProperty](#)、[GetObjectProperty](#)、[GetShortProperty](#)、[GetStringProperty](#)、[SetBooleanProperty](#)、[SetByteProperty](#)、[SetBytesProperty](#)、[SetCharProperty](#)、[SetDoubleProperty](#)、[SetFloatProperty](#)、[SetIntProperty](#)、[SetLongProperty](#)、[SetObjectProperty](#)、[SetShortProperty](#)、[SetStringProperty](#)

## ITextMessage

テキスト・メッセージとは、本体がストリングからなるメッセージです。

#### 継承の階層:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.ITextMessage
```

#### 関連資料

[テキスト・メッセージ](#)

テキスト・メッセージの本文には、ストリングが含まれています。

## ドット・ネットプロパティ

#### ドット・ネットプロパティの要約:

ドット・ネットプロパティ

[Text](#)

#### 説明

テキスト・メッセージの本文を形成するストリングを取得および設定します。

*Text* - テキストの取得および設定

#### インターフェース:

```
String Text
{
    get;
    set;
}
```

テキスト・メッセージの本文を形成するストリングを取得および設定します。

必要な場合、XMS がストリング内の文字をローカル・コード・ページに変換します。

#### 例外:

- XMSException

- `MessageNotReadableException`
- `MessageNotWritableException`
- `MessageEOFException`

## 継承されたプロパティおよびメソッド

以下のプロパティは、`IMessage` インターフェースから継承されています。

`JMSCorrelationID`、`JMSDeliveryMode`、`JMSDestination`、`JMSExpiration`、`JMSMessageID`、`JMSPriority`、`JMSRedelivered`、`JMSReplyTo`、`JMSTimestamp`、`JMSType`、`Properties`

以下のメソッドは、`IMessage` インターフェースから継承されています。

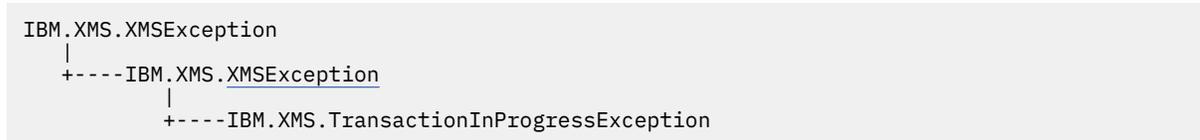
`clearBody`、`clearProperties`、`PropertyExists`

以下のメソッドは、`IPropertyContext` インターフェースから継承されています。

`GetBooleanProperty`、`GetByteProperty`、`GetBytesProperty`、`GetCharProperty`、`GetDoubleProperty`、`GetFloatProperty`、`GetIntProperty`、`GetLongProperty`、`GetObjectProperty`、`GetShortProperty`、`GetStringProperty`、`SetBooleanProperty`、`SetByteProperty`、`SetBytesProperty`、`SetCharProperty`、`SetDoubleProperty`、`SetFloatProperty`、`SetIntProperty`、`SetLongProperty`、`SetObjectProperty`、`SetShortProperty`、`SetStringProperty`

## TransactionInProgressException

継承の階層:



XMS がこの例外をスローするのは、トランザクションが進行中であるために無効になっている操作をアプリケーションが要求した場合です。

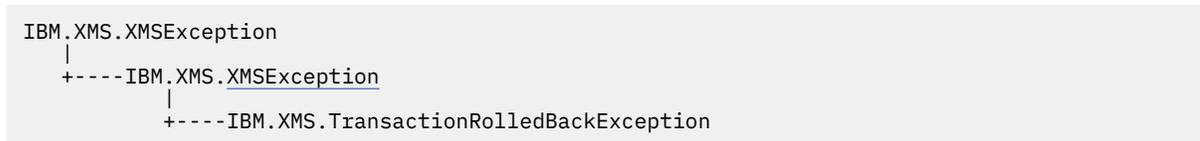
## 継承されたプロパティおよびメソッド

以下のメソッドは、`XMSEException` インターフェースから継承されています。

`GetErrorCode`、`GetLinkedException`

## TransactionRolledBackException

継承の階層:



XMS がこの例外をスローするのは、アプリケーションが現行のトランザクションをコミットするために `Session.commit()` を呼び出したにもかかわらず、その後このトランザクションがロールバックされた場合です。

## 継承されたプロパティおよびメソッド

以下のメソッドは、`XMSEException` インターフェースから継承されています。

`GetErrorCode`、`GetLinkedException`

## XMSEException

XMS がドット・ネット メソッドの呼び出しを処理しているときにエラーを検出すると、XMS は例外をスローします。例外とは、エラーに関する情報をカプセル化するオブジェクトのことです。

## 継承の階層:

```
System.Exception
|
+----IBM.XMS.XMSEException
```

XMS 例外にはさまざまなタイプがあり、XMSEException オブジェクトは例外の 1 タイプに過ぎません。ただし、XMSEException クラスは、その他の XMS 例外クラスのスーパークラスです。XMS は、XMSEException オブジェクト以外のタイプの例外では適切でない状態では、XMSEException オブジェクトをスローします。

## ドット・ネット プロパティ

### ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<a href="#">ErrorCode</a>	エラー・コードを取得します。
<a href="#">LinkedException</a>	例外のチェーン内の次の例外を取得します。

*ErrorCode* - エラー・コードの取得

### インターフェース:

```
public String ErrorCode
{
    get {return errorCode_;}
}
```

エラー・コードを取得します。

### 例外:

- XMSEException

*LinkedException* - リンク例外の取得

### インターフェース:

```
public Exception LinkedException
{
    get { return linkedException_;}
    set { linkedException_ = value;}
}
```

例外のチェーン内の次の例外を取得します。

チェーン内に次の例外がない場合、このメソッドは NULL を戻します。

### 例外:

- XMSEException

## XMSFactoryFactory

アプリケーションが管理対象オブジェクトを使用していない場合は、このクラスを使用して接続ファクトリー、キュー、およびトピックを作成します。

### 継承の階層:

なし

## ドット・ネット プロパティ

ドット・ネット プロパティの要約:

ドット・ネット プロパティ	説明
<a href="#">MetaData</a>	XMSFactoryFactory オブジェクトの接続タイプに該当するメタデータを取得します。

*Metadata* - メタデータの検索

インターフェース:

```
IConnectionMetaData MetaData
```

XMSFactoryFactory オブジェクトの接続タイプに該当するメタデータを取得します。

例外:

なし

## 方法

メソッドの要約:

メソッド	説明
<a href="#">CreateConnectionFactory</a>	宣言したタイプの ConnectionFactory オブジェクトを作成します。
<a href="#">CreateQueue</a>	メッセージング・サーバー内のキューを表すための Destination オブジェクトを作成します。
<a href="#">CreateTopic</a>	トピックを表すための Destination オブジェクトを作成します。
<a href="#">GetInstance</a>	XMSFactoryFactory のインスタンスを作成します。XMS アプリケーションは、XMSFactoryFactory オブジェクトを使用して、必要なタイプのプロトコルに適した ConnectionFactory オブジェクトへの参照を取得します。この結果、この ConnectionFactory オブジェクトは、対象のプロトコル・タイプに対してのみ接続経路を作成できます。

*CreateConnectionFactory* - 接続ファクトリーの作成

インターフェース:

```
IConnectionFactory CreateConnectionFactory();
```

宣言したタイプの ConnectionFactory オブジェクトを作成します。

パラメーター:

なし

戻り値:

ConnectionFactory オブジェクト。

例外:

- XMSException

*CreateQueue* - キューの作成

インターフェース:

```
IDestination CreateQueue(String name);
```

メッセージング・サーバー内のキューを表すための Destination オブジェクトを作成します。

このメソッドは、メッセージング・サーバー内にキューを作成しません。アプリケーションがこのメソッドを呼び出すためには、その前にキューを作成する必要があります。

**パラメーター:**

**name (入力)**

キューの名前をカプセル化している String オブジェクト、またはキューを識別する Uniform Resource Identifier (URI) をカプセル化している String オブジェクト。

**戻り値:**

キューを表す Destination オブジェクト。

**例外:**

- XMSEException

*CreateTopic* - トピックの作成

**インターフェース:**

```
IDestination CreateTopic(String name);
```

トピックを表すための Destination オブジェクトを作成します。

**パラメーター:**

**name (入力)**

トピックの名前をカプセル化している String オブジェクト、またはトピックを識別する Uniform Resource Identifier (URI) をカプセル化している String オブジェクト。

**戻り値:**

トピックを表す Destination オブジェクト。

**例外:**

- XMSEException

*GetInstance* - XMSFactoryFactory のインスタンスの取得

**インターフェース:**

```
static XMSFactoryFactory GetInstance(int connectionType);
```

XMSFactoryFactory のインスタンスを作成します。XMS アプリケーションは、XMSFactoryFactory オブジェクトを使用して、必要なタイプのプロトコルに適した ConnectionFactory オブジェクトへの参照を取得します。この結果、この ConnectionFactory オブジェクトは、対象のプロトコル・タイプに対してのみ接続経路を作成できます。

**パラメーター:**

**connectionType (入力)**

ConnectionFactory オブジェクトによって接続経路が作成される接続のタイプ

- XMSC.CT\_WPM
- XMSC.CT\_RTT
- XMSC.CT\_WMQ

**戻り値:**

宣言した接続タイプ専用の XMSFactoryFactory オブジェクト。

**例外:**

- NotSupportedException

## XMS オブジェクトのプロパティー

この節章では、XMS で定義したオブジェクトのプロパティーについて説明します。

節章には、以下のトピック節が含まれています。

- [182 ページの『Connection のプロパティ』](#)
- [183 ページの『ConnectionFactory のプロパティ』](#)
- [189 ページの『ConnectionMetaData のプロパティ』](#)
- [189 ページの『Destination のプロパティ』](#)
- [192 ページの『InitialContext のプロパティ』](#)
- [193 ページの『Message のプロパティ』](#)
- [198 ページの『MessageConsumer のプロパティ』](#)
- [198 ページの『MessageProducer のプロパティ』](#)
- [198 ページの『Session のプロパティ』](#)

各トピック節には、指定された型のオブジェクトのプロパティがリストで表示され、各プロパティの簡略説明が記載されています。

この節章には、各プロパティの定義をまとめた [198 ページの『プロパティ定義』](#) というトピック節も含まれています。

この節章で取り上げているオブジェクトに関する独自のプロパティをアプリケーションで定義すると、エラーは発生しませんが、予測できない結果が発生する可能性があります。

**注:** このセクションのプロパティ名および値は、`XMSC.NAME` という形式で表示されます。これは、C および C++ に使用される形式です。ただし、ドット・ネットでは、プロパティ名の形式は、`XMSC.NAME` または `XMSC_NAME` のいずれかにすることができます。使用方法によっては、以下のようになります。

- プロパティを指定している場合、以下の例のように、プロパティ名は `XMSC.NAME` の形式でなければなりません。

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- スtringを指定している場合、以下の例のように、プロパティ名は `XMSC_NAME` の形式でなければなりません。

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

ドット・ネットでは、プロパティ名と値は XMSC クラスの定数として提供されます。これらの定数は String を識別し、任意の XMS ドット・ネット アプリケーションによって使用されます。これらの事前定義定数を使用する場合、プロパティ名と値の形式は `XMSC.NAME` となるため、例えば `XMSC_USERID` ではなく、`XMSC.USERID` を使用します。

データ・タイプは、C/C++ で使用される形式にもなっています。ドット・ネットの対応する値については、[46 ページの『ドット・ネットのデータ型』](#) を参照してください。

## 関連概念

[ユーザー独自のアプリケーションの作成](#)

ユーザー独自のアプリケーションをビルドする方法は、サンプル・アプリケーションをビルドする場合と同様です。

## 関連資料

[ドット・ネットインターフェース](#)

このトピック節では、ドット・ネット クラスのインターフェースとそのプロパティおよびメソッドについて説明します。

## Connection のプロパティ

Connection オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
234 ページの『 <a href="#">XMSC_WMQ_RESOLVED_QUEUE_MANAGER</a> 』	このプロパティは、接続先のキュー・マネージャーの名前を取得するために使用します。
234 ページの『 <a href="#">XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID</a> 』	接続後に、このプロパティにはキュー・マネージャーの ID が設定されます。
<a href="#">XMSC_WPM_CONNECTION_PROTOCOL</a>	メッセージング・エンジンへの接続に使用される通信プロトコルです。このプロパティは読み取り専用です。
<a href="#">XMSC_WPM_HOST_NAME</a>	アプリケーションの接続先となるメッセージング・エンジンが収容されているシステムのホスト名または IP アドレスです。このプロパティは読み取り専用です。
<a href="#">XMSC_WPM_ME_NAME</a>	アプリケーションの接続先となるメッセージング・エンジンの名前です。このプロパティは読み取り専用です。
<a href="#">XMSC_WPM_PORT</a>	アプリケーションの接続先となるメッセージング・エンジンによって listen されるポートの数です。このプロパティは読み取り専用です。

Connection オブジェクトには、接続経路を作成するときに使用した接続ファクトリーのプロパティから導出した読み取り専用プロパティもあります。これらのプロパティの導出元は、接続経路作成時に設定された接続ファクトリー・プロパティだけでなく、未設定だったプロパティのデフォルト値の場合もあります。これらのプロパティは、アプリケーションの接続先になっているメッセージング・サーバーのタイプに該当するプロパティに限定されます。プロパティの名前は、接続ファクトリー・プロパティの名前と同じです。

## ConnectionFactory のプロパティ

ConnectionFactory オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
211 ページの『 <a href="#">XMSC_ASYNC_EXCEPTIONS</a> 』	このプロパティは、XMS が <code>ExceptionListener</code> への通知を、接続が切断されたときのみ行うか、または XMS API 呼び出しに対して非同期に例外が発生したときに行うかを決定します。このプロパティは、 <code>ExceptionListener</code> が登録されているこの <code>ConnectionFactory</code> から作成されたすべての接続に適用されます。
<a href="#">XMSC_CLIENT_ID</a>	接続のクライアント ID です。
<a href="#">XMSC_CONNECTION_TYPE</a>	アプリケーションの接続先となるメッセージング・サーバーのタイプです。
<a href="#">XMSC_PASSWORD</a>	アプリケーションがメッセージング・サーバーに接続しようとしているときに、このアプリケーションを認証するために使用するパスワードです。
216 ページの『 <a href="#">XMSC_RTT_BROKER_PING_INTERVAL</a> 』	XMS.NET がリアルタイム・メッセージング・サーバーへの接続を検査することでアクティビティの検出を行うまでの時間間隔 (ミリ秒単位)。
<a href="#">XMSC_RTT_CONNECTION_PROTOCOL</a>	ブローカーへのリアルタイム接続に使用される通信プロトコルです。
<a href="#">XMSC_RTT_HOST_NAME</a>	ブローカーを実行するシステムのホスト名または IP アドレスです。

表 29. ConnectionFactory のプロパティ (続き)	
プロパティ名	説明
<u>XMSC_RTT_LOCAL_ADDRESS</u>	ブローカーへのリアルタイム接続に使用するローカル・ネットワーク・インターフェースのホスト名または IP アドレスです。
<u>XMSC_RTT_MULTICAST</u>	接続ファクトリーまたは宛先のマルチキャスト設定です。
<u>XMSC_RTT_PORT</u>	ブローカーが着信要求を listen するポートの数です。
<u>XMSC_USERID</u>	アプリケーションがメッセージング・サーバーに接続しようとしているときに、このアプリケーションを認証するために使用するユーザー ID です。
<u>XMSC_WMQ_BROKER_CONTROLQ</u>	ブローカーが使用する制御キューの名前です。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
<u>XMSC_WMQ_BROKER_PUBQ</u>	ブローカーによってモニターされているキューの名前で、このキューでは、アプリケーションが発行したメッセージをアプリケーション自体が送信します。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
<u>XMSC_WMQ_BROKER_QMGR</u>	ブローカーの接続先となるキュー・マネージャーの名前です。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
<u>XMSC_WMQ_BROKER_SUBQ</u>	非永続メッセージ・コンシューマー用のサブスクライバー・キューの名前です。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。

表 29. ConnectionFactory のプロパティ (続き)	
プロパティ名	説明
<u>XMSC_WMQ_BROKER_VERSION</u>	<p>接続または宛先に合わせてアプリケーションが使用するブローカーのタイプです。</p> <p><b>注:</b> このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。</p>
221 ページの『 <u>XMSC_WMQ_CCDURL</u> 』	クライアント・チャンネル定義テーブルを含むファイルの名前および場所を識別すると同時に、そのファイルへのアクセス方法も指定する、URL (Uniform Resource Locator)。
<u>XMSC_WMQ_CHANNEL</u>	接続に使用するチャンネルの名前です。
221 ページの『 <u>XMSC_WMQ_CLIENT_RECONNECT_OPTIONS</u> 』	このプロパティでは、このファクトリーで作成する新しい接続のクライアント再接続オプションを指定します。
222 ページの『 <u>XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT</u> 』	このプロパティでは、クライアント接続が再接続を試みる期間を秒単位で指定します。
<u>XMSC_WMQ_CONNECTION_MODE</u>	アプリケーションがキュー・マネージャーに接続する場合のモードです。
223 ページの『 <u>XMSC_WMQ_CONNECTION_NAME_LIST</u> 』	このプロパティでは、接続で障害が発生した後にクライアントが再接続を試みる対象のホストを指定します。
<u>XMSC_WMQ_FAIL_IF QUIESCE</u>	アプリケーションの接続先のキュー・マネージャーが静止状態である場合、特定のメソッドの呼び出しが失敗するかどうかを表します。
<u>XMSC_WMQ_HOST_NAME</u>	キュー・マネージャーを実行するシステムのホスト名または IP アドレスです。
<u>XMSC_WMQ_LOCAL_ADDRESS</u>	キュー・マネージャーへの接続の場合、このプロパティは、使用するローカル・ネットワーク・インターフェース、または使用するローカル・ポート (1 つまたは一定範囲)、あるいはその両方を指定します。
<u>XMSC_WMQ_MESSAGE_SELECTION</u>	<p>メッセージの選択を XMS クライアントで行うか、ブローカーで行うかを判別します。</p> <p><b>注:</b> このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。</p>
<u>XMSC_WMQ_MSG_BATCH_SIZE</u>	<p>非同期のメッセージ配信機能を使用する場合に 1 バッチ内のキューから取り出すメッセージの最大数を表します。</p> <p><b>注:</b> このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。</p>

表 29. ConnectionFactory のプロパティ (続き)	
プロパティ名	説明
<u>XMSC_WMQ_POLLING_INTERVAL</u>	セッション内の各メッセージ・リスナーのキューに適切なメッセージがない場合、この値は、各メッセージ・リスナーがそのキューから再度メッセージを読み取ろうとするまでに経過する最大の時間間隔 (ミリ秒) になります。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
231 ページの 『 <u>XMSC_WMQ_PROVIDER_VERSION</u> 』	アプリケーションの接続先のキュー・マネージャーのバージョン、リリース、モディフィケーション・レベル、およびフィックスパック。
<u>XMSC_WMQ_PORT</u>	キュー・マネージャーが着信要求を listen するポートの数です。
<u>XMSC_WMQ_PUB_ACK_INTERVAL</u>	XMS クライアントがブローカーから肯定応答を要求するまでにパブリッシャーによって発行されるメッセージの数です。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの XMSC_WMQ_PROVIDER_VERSION プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
227 ページの 『 <u>XMSC_WMQ_PUT_ASYNC_ALLOWED</u> 』	このプロパティは、メッセージ・プロデューサーが、非同期書き込みを使用して、この宛先にメッセージを送信できるかどうかを決定します。
<u>XMSC_WMQ_QMGR_CCSD</u>	メッセージ・キュー・インターフェース (MQI) で定義された文字データのフィールドが XMS クライアントと IBM WebSphere MQ クライアントの間で交換されるコード化文字セットまたはコード・ページの ID (CCSID)。
<u>XMSC_WMQ_QUEUE_MANAGER</u>	接続先となるキュー・マネージャーの名前です。
<u>XMSC_WMQ_RECEIVE_EXIT</u>	実行するチャンネル受信出口を示します。
<u>XMSC_WMQ_RECEIVE_EXIT_INIT</u>	チャンネル受信出口が呼び出されたときにこの出口に渡されるユーザー・データです。
<u>XMSC_WMQ_SECURITY_EXIT</u>	チャンネル・セキュリティー出口を示します。
<u>XMSC_WMQ_SECURITY_EXIT_INIT</u>	チャンネル・セキュリティー出口を呼び出した場合にこの出口に渡されるユーザー・データです。
236 ページの 『 <u>XMSC_WMQ_SEND_CHECK_COUNT</u> 』	単一の未処理の XMS セッション内での、非同期書き込みエラーの検査の間に許可する Send 呼び出しの数。
<u>XMSC_WMQ_SEND_EXIT</u>	チャンネル送信出口を示します。
<u>XMSC_WMQ_SEND_EXIT_INIT</u>	複数のチャンネル送信出口を呼び出した場合にこれらの出口に渡されるユーザー・データです。

表 29. ConnectionFactory のプロパティ (続き)	
プロパティ名	説明
236 ページの 『XMSC_WMQ_SHARE_CONV_ALLOWED』	チャンネル定義が一致する場合に、クライアント接続が、同じプロセスから同じキュー・マネージャーへの他の最上位 XMS 接続とソケットを共有できるかどうか。このプロパティは、アプリケーション開発、保守、または操作に必要な場合に、別のソケットの接続を完全に分離できるようにするために提供されています。
XMSC_WMQ_SSL_CERT_STORES	キュー・マネージャーとの SSL 接続で使用される証明書取り消しリスト (CRL) を保持するサーバーの位置。
XMSC_WMQ_SSL_CIPHER_SPEC	キュー・マネージャーとのセキュア接続で使用する CipherSpec の名前。
XMSC_WMQ_SSL_CIPHER_SUITE	キュー・マネージャーへの SSL 接続または TLS 接続で使用される CipherSuite の名前。セキュア接続のネゴシエーションで使用されるプロトコルは、指定されている CipherSuite によって異なります。
XMSC_WMQ_SSL_CRYPT_HW	クライアント・システムに接続されている暗号ハードウェアに関する構成詳細情報。
XMSC_WMQ_SSL_FIPS_REQUIRED	このプロパティの値は、アプリケーションが非 FIPS 準拠の暗号スイートを使用できるかどうかを決定します。このプロパティが true (真) に設定されている場合、クライアント/サーバー接続には FIPS アルゴリズムだけが使用されます。
XMSC_WMQ_SSL_KEY_REPOSITORY	鍵および証明書が保存されている鍵データベース・ファイルの位置。
XMSC_WMQ_SSL_KEY_RESETCOUNT	KeyResetCount は、秘密鍵の再ネゴシエーションが実行されるまで、1 つの SSL 会話の中で送受信される暗号化されていないデータの合計バイト数を表します。
XMSC_WMQ_SSL_PEER_NAME	キュー・マネージャーとの SSL 接続で使用されるピア名。
XMSC_WMQ_SYNCPOINT_ALL_GETS	すべてのメッセージを同期点制御の対象範囲内のキューから取り出す必要があるかどうかを示します。
243 ページの『XMSC_WMQ_TARGET_CLIENT』	
XMSC_WMQ_TEMP_Q_PREFIX	アプリケーションが XMS 一時キューを作成するときに作成される IBM WebSphere MQ 動的キューの名前を形成するために使用される接頭部。
XMSC_WMQ_TEMP_TOPIC_PREFIX	一時トピックの作成時に、XMS は「TEMP/TEMPTOPICPREFIX/unique_id」の形式のトピック・ストリングを生成します。このプロパティがデフォルト値のままである場合は、単に「TEMP/unique_id」を生成します。空以外の値を指定すると、この接続で作成された一時トピックへのサブスクリバの管理対象キューを作成するために、特定のモデル・キューを定義できます。
XMSC_WMQ_TEMPORARY_MODEL	アプリケーションが XMS 一時キューを作成するときに動的キューの作成元となる IBM WebSphere MQ モデル・キューの名前。

表 29. <i>ConnectionFactory</i> のプロパティ (続き)	
プロパティ名	説明
<a href="#">XMSC_WPM_BUS_NAME</a>	接続ファクトリーの場合は、アプリケーションの接続先となるサービス統合バスの名前であり、宛先の場合は、その宛先が存在するサービス統合バスの名前です。
<a href="#">XMSC_WPM_CONNECTION_PROXIMITY</a>	接続に対する接続接近性の設定です。
<a href="#">XMSC_WPM_DUR_SUB_HOME</a>	ある接続または宛先に対するすべての永続サブスクリプションが管理対象になっているメッセージング・エンジンの名前です。
<a href="#">XMSC_WPM_LOCAL_ADDRESS</a>	サービス統合バスへの接続の場合、このプロパティは、使用するローカル・ネットワーク・インターフェース、または使用するローカル・ポート (1つまたは一定範囲)、あるいはその両方を指定します。
<a href="#">XMSC_WPM_NON_PERSISTENT_MAP</a>	接続を使用して送信される非永続メッセージの信頼性レベルです。
<a href="#">XMSC_WPM_PERSISTENT_MAP</a>	接続を使用して送信される永続メッセージの信頼性レベルです。
<a href="#">XMSC_WPM_PROVIDER_ENDPOINTS</a>	ブートストラップ・サーバーの1つ以上のエンドポイント・アドレスの列です。
<a href="#">XMSC_WPM_TARGET_GROUP</a>	メッセージング・エンジンのターゲット・グループの名前です。
<a href="#">XMSC_WPM_TARGET_SIGNIFICANCE</a>	メッセージング・エンジンのターゲット・グループの重要度です。
<a href="#">XMSC_WPM_TARGET_TRANSPORT_CHAIN</a>	アプリケーションがメッセージング・エンジンに接続するために使用する必要があるインバウンド・トランスポート・チェーンの名前です。
<a href="#">XMSC_WPM_TARGET_TYPE</a>	メッセージング・エンジンのターゲット・グループのタイプです。
<a href="#">XMSC_WPM_TEMP_Q_PREFIX</a>	アプリケーションがXMS一時キューを作成するとサービス統合バスで作成される一時キューの名前を構成する場合に使用されるプレフィックスです。
<a href="#">XMSC_WPM_TEMP_TOPIC_PREFIX</a>	アプリケーションによって作成される一時トピックの名前を構成する場合に使用されるプレフィックスです。

## 関連概念

### [ConnectionFactory オブジェクトと Connection オブジェクト](#)

*ConnectionFactory* オブジェクトには、アプリケーションが *Connection* オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、*Connection* オブジェクトを使用して *Session* オブジェクトを作成します。

### [WebSphere サービス統合バスへの接続](#)

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

### [IBM WebSphere MQ キュー・マネージャーへのセキュア接続](#)

XMS .NET アプリケーションが、IBM WebSphere MQ キュー・マネージャーとのセキュア接続を確立できるようにするには、関係するプロパティが *ConnectionFactory* オブジェクトで定義されていることが必要です。

### [WebSphere サービス統合バス メッセージング・エンジンとのセキュア接続](#)

XMS .NET アプリケーションが WebSphere サービス統合バス・メッセージング・エンジンとのセキュア接続を確立できるようにするには、関係するプロパティが `ConnectionFactory` オブジェクトで定義されている必要があります。

#### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

#### 関連タスク

##### 管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする `ConnectionFactory` および `Destination` オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

#### 関連資料

##### 管理対象 `ConnectionFactory` オブジェクトの必須プロパティ

アプリケーションが接続ファクトリーを作成する場合には、メッセージング・サーバーへの接続を作成するために多くのプロパティを定義する必要があります。

## ConnectionMetaData のプロパティ

`ConnectionMetaData` オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
<a href="#">XMSC_JMS_MAJOR_VERSION</a>	XMS が準拠する JMS 仕様のメジャー・バージョン番号です。このプロパティは読み取り専用です。
<a href="#">XMSC_JMS_MINOR_VERSION</a>	XMS が準拠する JMS 仕様のマイナー・バージョン番号です。このプロパティは読み取り専用です。
<a href="#">XMSC_JMS_VERSION</a>	XMS が準拠する JMS 仕様のバージョン ID です。このプロパティは読み取り専用です。
<a href="#">XMSC_MAJOR_VERSION</a>	XMS クライアントのバージョン番号です。このプロパティは読み取り専用です。
<a href="#">XMSC_MINOR_VERSION</a>	XMS クライアントのリリース番号です。このプロパティは読み取り専用です。
<a href="#">XMSC_PROVIDER_NAME</a>	XMS クライアントのプロバイダーです。このプロパティは読み取り専用です。
<a href="#">XMSC_VERSION</a>	XMS クライアントのバージョン ID です。このプロパティは読み取り専用です。

## Destination のプロパティ

`Destination` オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
<a href="#">XMSC_DELIVERY_MODE</a>	宛先に送信されたメッセージの送達モードです。
<a href="#">XMSC_PRIORITY</a>	宛先に送信されたメッセージの優先順位です。
<a href="#">XMSC_RTT_MULTICAST</a>	接続ファクトリーまたは宛先のマルチキャスト設定です。
<a href="#">XMSC_TIME_TO_LIVE</a>	宛先に送信されたメッセージの存続時間です。

表 31. Destination のプロパティ (続き)	
プロパティ名	説明
<u>XMSC_WMQ_BROKER_VERSION</u>	接続または宛先に合わせてアプリケーションが使用するローカーのタイプです。
<u>XMSC_WMQ_CC SID</u>	XMS クライアントがメッセージを宛先に転送する場合、メッセージの本文中の一連の文字データに設定されるコード化文字セットの ID (CCSID)、つまりコード・ページです。
<u>XMSC_WMQ_DUR_SUBQ</u>	宛先からメッセージを受信している永続サブスクライバー用のサブスクライバー・キューの名前です。  注: このプロパティは、IBM Message Service Client for .NET バージョン 2.0 で使用できますが、接続ファクトリーの <u>XMSC_WMQ_PROVIDER_VERSION</u> プロパティが 7 より前のバージョン番号に設定されている場合を除き、IBM WebSphere MQ バージョン 7.0 キュー・マネージャーに接続されているアプリケーションでは無効になります。
<u>XMSC_WMQ_ENCODING</u>	XMS クライアントがメッセージを宛先に転送する場合、メッセージの本文中の数値データをどのように表記するかを表します。
<u>XMSC_WMQ_FAIL_IF_QUIESCE</u>	アプリケーションの接続先のキュー・マネージャーが静止状態である場合、特定のメソッドの呼び出しが失敗するかどうかを表します。
224 ページの『 <u>XMSC_WMQ_MESSAGE_BODY</u> 』	このプロパティは、XMS アプリケーションが IBM WebSphere MQ メッセージの MQRFH2 をメッセージ・ペイロードの一部として (つまり、メッセージ本体の一部として) 処理するかどうかを決定します。
225 ページの『 <u>XMSC_WMQ_MQMD_MESSAGE_CONTEXT</u> 』	XMS アプリケーションによって設定されるメッセージ・コンテキストのレベルを決定します。このプロパティが有効となるためには、アプリケーションが適切なコンテキスト権限を持って実行されていなければなりません。
226 ページの『 <u>XMSC_WMQ_MQMD_READ_ENABLED</u> 』	このプロパティは、XMS アプリケーションが MQMD フィールドの値を抽出できるかどうかを決定します。
226 ページの『 <u>XMSC_WMQ_MQMD_WRITE_ENABLED</u> 』	このプロパティは、XMS アプリケーションが MQMD フィールドの値を書き込むことができるかどうかを決定します。
227 ページの『 <u>XMSC_WMQ_READ_AHEAD_ALLOWED</u> 』	このプロパティは、メッセージ・コンシューマーとキュー・ブラウザーが、先行読み取りを使用して、非永続の非トランザクション・メッセージを受信する前に、この宛先から内部バッファにこれらのメッセージを取得できるかどうかを決定します。
228 ページの『 <u>XMSC_WMQ_READ_AHEAD_CLOSE_POLICY</u> 』	このプロパティは、非同期メッセージ・リスナーに配信されているメッセージについて、メッセージ・コンシューマーがクローズされたときに、内部先行読み取りバッファ内のメッセージがどうなるかを決定します。
233 ページの『 <u>XMSC_WMQ_RECEIVE_CC SID</u> 』	キュー・マネージャー・メッセージ変換のターゲット CCSID を設定する宛先プロパティ。 <u>XMSC_WMQ_RECEIVE_CONVERSION</u> が <u>WMQ_RECEIVE_CONVERSION_QMGR</u> に設定されていなければ、この値は無視されます。

表 31. Destination のプロパティ (続き)	
プロパティ名	説明
233 ページの『XMSC_WMQ_RECEIVE_CONVERSION』	キュー・マネージャーによりデータ変換を実行するかどうかを決定する宛先プロパティ。
XMSC_WMQ_TARGET_CLIENT	宛先に送信されるメッセージに MQRFH2 ヘッダーを付けるかどうかを示します。
XMSC_WMQ_TEMP_TOPIC_PREFIX	一時トピックの作成時に、XMS は「TEMP/TEMPTOPICPREFIX/unique_id」の形式のトピック・ストリングを生成します。このプロパティがデフォルト値のままである場合は、単に「TEMP/unique_id」を生成します。空以外の値を指定すると、この接続で作成された一時トピックへのサブスクライバーの管理対象キューを作成するために、特定のモデル・キューを定義できます。
XMSC_WPM_BUS_NAME	接続ファクトリーの場合は、アプリケーションの接続先となるサービス統合バスの名前であり、宛先の場合は、その宛先が存在するサービス統合バスの名前です。
XMSC_WPM_TOPIC_SPACE	トピックが収容されているトピック・スペースの名前です。

## 関連概念

### ConnectionFactory オブジェクトと Connection オブジェクト

ConnectionFactory オブジェクトには、アプリケーションが Connection オブジェクトを作成するときに使用するテンプレートがあります。アプリケーションは、Connection オブジェクトを使用して Session オブジェクトを作成します。

### WebSphere サービス統合バスへの接続

XMS アプリケーションは、TCP/IP 接続を直接使用するか、HTTP over TCP/IP を使用することにより、WebSphere サービス統合バスに接続できます。

### 宛先

XMS アプリケーションは、送信対象メッセージの宛先と受信対象メッセージの送信元を指定するときに Destination オブジェクトを使用します。

### 宛先のワイルドカード

XMS では、宛先のワイルドカードがサポートされているため、必要な場所にワイルドカードを挿入して突き合わせを行うことができます。XMS が処理できるサーバー・タイプごとに、ワイルドカード・スキームは異なります。

### トピック URI

トピック URI はトピック名を指定します。また、オプションでトピックのプロパティ (複数可) を指定することもできます。

### キュー URI

キューの URI は、キューの名前を指定します。また、オプションでキューのプロパティ (複数可) を指定することもできます。

### 一時宛先

XMS アプリケーションは一時宛先を作成および使用できます。

### 管理対象オブジェクトのプロパティ・マッピング

アプリケーションを使用可能にして、IBM WebSphere MQ JMS と WebSphere ApplicationServer の接続ファクトリー・オブジェクト定義および宛先オブジェクト定義を使用するには、これらの定義から取り出したプロパティを、定義に対応する XMS プロパティで、かつ XMS 接続ファクトリーおよび宛先に設定できるプロパティにマップする必要があります。

## 関連タスク

管理対象オブジェクトの作成

メッセージング・サーバーへの接続を作成するために XMS アプリケーションが必要とする ConnectionFactory および Destination オブジェクト定義は、適切な管理ツールを使用して作成する必要があります。

### 関連資料

管理対象 Destination オブジェクトの必須プロパティ

宛先を作成するアプリケーションは、管理対象 Destination オブジェクトで複数のプロパティを設定する必要があります。

## InitialContext のプロパティ

InitialContext オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

表 32. InitialContext のプロパティ	
プロパティ名	説明
<a href="#">XMSC_IC_URL</a>	LDAP コンテキストや FileSystem コンテキストの場合は、管理対象オブジェクトを収容しているリポジトリのアドレスです。

表 33. InitialContext のプロパティ	
プロパティ名	説明
<a href="#">XMSC_IC_PROVIDER_URL</a>	JNDI ネーミング・ディレクトリーの位置を指定するために使用します。その位置を指定すれば、COS ネーミング・サービスが Web サービスと同じサーバーに存在する必要はなくなります。
<a href="#">XMSC_IC_SECURITY_AUTHENTICATION</a>	Java コンテキスト・インターフェース SECURITY_AUTHENTICATION に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。
<a href="#">XMSC_IC_SECURITY_CREDENTIALS</a>	Java コンテキスト・インターフェース SECURITY_CREDENTIALS に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。
<a href="#">XMSC_IC_SECURITY_PRINCIPAL</a>	Java コンテキスト・インターフェース SECURITY_PRINCIPAL に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。
<a href="#">XMSC_IC_SECURITY_PROTOCOL</a>	Java コンテキスト・インターフェース SECURITY_PROTOCOL に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。
<a href="#">XMSC_IC_URL</a>	LDAP コンテキストや FileSystem コンテキストの場合は、管理対象オブジェクトを収容しているリポジトリのアドレスです。COS ネーミング・コンテキストの場合は、ディレクトリー内のオブジェクトを検索する Web サービスのアドレスです。

### 関連概念

InitialContext プロパティ

InitialContext コンストラクターのパラメーターには Uniform Resource Indicator (URI) で指定される、管理対象オブジェクトのリポジトリのロケーションが含まれます。アプリケーションがリポジトリへの接続を確立するためには、URI に含まれる情報より多くの情報を指定することが必要な場合があります。

[XMS 初期コンテキストの URI フォーマット](#)

管理対象オブジェクトのリポジトリのロケーションは、Uniform Resource Indicator (URI) で指定します。URI のフォーマットは、コンテキストのタイプにより異なります。

#### 管理対象オブジェクトの検索

XMS は、InitialContext オブジェクトの作成時に指定されたアドレス、または InitialContext プロパティに指定されているアドレスを使用して、リポジトリから管理対象オブジェクトを取り出します。

#### 関連タスク

##### InitialContext オブジェクト

アプリケーションは、管理対象オブジェクト・リポジトリへの接続を作成するために使用される初期コンテキストを作成して、必要な管理対象オブジェクトを取得する必要があります。

## Message のプロパティ

Message オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
<a href="#">JMS_IBM_CHARACTER_SET</a>	XMS クライアントがメッセージを目的の宛先に転送する場合、メッセージの本文中の文字データ・ストリングが設定されるコード化文字セットの ID (CCSID)、またはコード・ページ。XMS では、このプロパティには数値が指定され、CCSID にマップされます。ただし、このプロパティは JMS プロパティに基づいているため、ストリング型の値を持ち、この数値 CCSID を表す Java 文字セットにマップされます。
<a href="#">JMS_IBM_ENCODING</a>	XMS クライアントがメッセージを目的の宛先に転送する場合、メッセージの本文中の数値データをどのように表記するかを表します。
<a href="#">JMS_IBM_EXCEPTIONMESSAGE</a>	メッセージが例外の宛先に送信された理由を説明するテキストです。このプロパティは読み取り専用です。
<a href="#">JMS_IBM_EXCEPTIONPROBLEMDESTINATION</a>	メッセージが例外の宛先に送信される前に、そのメッセージが存在した宛先の名前です。
<a href="#">JMS_IBM_EXCEPTIONREASON</a>	メッセージが例外の宛先に送信された理由を示す理由コードです。
<a href="#">JMS_IBM_EXCEPTIONTIMESTAMP</a>	メッセージが例外の宛先に送信された時刻です。
<a href="#">JMS_IBM_FEEDBACK</a>	レポート・メッセージの種類を示すコードです。
<a href="#">JMS_IBM_FORMAT</a>	メッセージ内にあるアプリケーション・データの種類です。
<a href="#">JMS_IBM_LAST_MSG_IN_GROUP</a>	メッセージがメッセージ・グループ内の最後のメッセージであるかどうかを示します。
<a href="#">JMS_IBM_MSGTYPE</a>	メッセージのタイプ。
<a href="#">JMS_IBM_PUTAPPLTYPE</a>	メッセージを送信したアプリケーションのタイプです。
<a href="#">JMS_IBM_PUTDATE</a>	メッセージが送信された日付です。
<a href="#">JMS_IBM_PUTTIME</a>	メッセージが送信された時刻です。
<a href="#">JMS_IBM_REPORT_COA</a>	「到着時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。

表 34. Message のプロパティ (続き)	
プロパティ名	説明
<u>JMS_IBM_REPORT_COD</u>	「配信時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。
<u>JMS_IBM_REPORT_DISCARD_MSG</u>	メッセージを目的の宛先に配信できない場合に、そのメッセージを廃棄することを要求します。
<u>JMS_IBM_REPORT_EXCEPTION</u>	例外レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。
<u>JMS_IBM_REPORT_EXPIRATION</u>	期限満了レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。
<u>JMS_IBM_REPORT_NAN</u>	否定アクション通知レポート・メッセージを要求します。
<u>JMS_IBM_REPORT_PAN</u>	肯定アクション通知レポート・メッセージを要求します。
<u>JMS_IBM_REPORT_PASS_CORREL_ID</u>	任意のレポート・メッセージまたは応答メッセージの相関 ID を元のメッセージの相関 ID と同じにするという要求です。
<u>JMS_IBM_REPORT_PASS_MSG_ID</u>	任意のレポート・メッセージまたは応答メッセージのメッセージ ID を元のメッセージのメッセージ ID と同じにするという要求です。
<u>JMS_IBM_RETAIN</u>	このプロパティを設定すると、キュー・マネージャーは、メッセージを保存パブリケーションとして扱うように指定されます。
<u>JMS_IBM_SYSTEM_MESSAGEID</u>	サービス統合バスの内部でメッセージを一意的に識別する ID です。このプロパティは読み取り専用です。
<u>JMSX_APPID</u>	メッセージを送信したアプリケーションの名前です。
<u>JMSX_DELIVERY_COUNT</u>	メッセージ配信の試行回数です。
<u>JMSX_GROUPID</u>	メッセージが属するメッセージ・グループの ID です。
<u>JMSX_GROUPSEQ</u>	メッセージ・グループ内にあるメッセージのシーケンス番号です。
<u>JMSX_USERID</u>	メッセージを送信したアプリケーションに関連付けられているユーザー ID です。

## JMS\_IBM\_MQMD\* プロパティ

IBM Message Service Client for .NET では、API を使用して、クライアント・アプリケーションによる MQMD フィールドの読み取り/書き込みが可能です。また、MQ メッセージ・データにアクセスすることもできます。デフォルトでは、MQMD へのアクセスは無効になっており、Destination のプロパティ `XMSC_WMQ_MQMD_WRITE_ENABLED` と `XMSC_WMQ_MQMD_READ_ENABLED` を使用してアプリケーションで明示的に有効にする必要があります。これらの 2 つのプロパティは互いに独立しています。

StrucId と Version を除く MQMD フィールドはすべて、追加の Message オブジェクト・プロパティとして公開され、JMS\_IBM\_MQMD というプレフィックスが付けられます。

JMS\_IBM\_MQMD\* プロパティは、上の表で取り上げられている他のプロパティ (JMS\_IBM\* など) より優先されます。

### メッセージの送信

StrucId と Version を除くすべての MQMD フィールドが表示されます。これらのプロパティは MQMD フィールドのみを参照しています。このフィールドでは、MQMD ヘッダーと MQRFH2 ヘッダーの両方でプロパティが発生し、MQRFH2 のバージョンは設定も抽出もされません。JMS\_IBM\_MQMD\_BackoutCount を除き、これらのすべてのプロパティを設定できます。JMS\_IBM\_MQMD\_BackoutCount に設定された値はすべて無視されます。

プロパティが最大長を持っていて、長過ぎる値が提供された場合、その値は切り捨てられます。

特定のプロパティでは、Destination オブジェクトで XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT プロパティも設定する必要があります。このプロパティが有効となるためには、アプリケーションが適切なコンテキスト権限を持って実行されていなければなりません。XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT を適切な値に設定しないと、プロパティ値は無視されます。XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT を適切な値に設定しても、キュー・マネージャーに対して十分なコンテキスト権限がない場合は、例外が発行されます。XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT の特定の値が必要なプロパティは以下のとおりです。

以下のプロパティでは、XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT を XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT または XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT に設定する必要があります。

- JMS\_IBM\_MQMD\_UserIdentifier
- JMS\_IBM\_MQMD\_AccountingToken
- JMS\_IBM\_MQMD\_ApplIdentityData

以下のプロパティでは、XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT を XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT に設定する必要があります。

- JMS\_IBM\_MQMD\_PutApplType
- JMS\_IBM\_MQMD\_PutApplName
- JMS\_IBM\_MQMD\_PutDate
- JMS\_IBM\_MQMD\_PutTime
- JMS\_IBM\_MQMD\_ApplOriginData

### メッセージの受信

XMSC\_WMQ\_MQMD\_READ\_ENABLED プロパティが true に設定されていれば、メッセージを作成するアプリケーションで設定されている実際のプロパティに関係なく、受信するメッセージでこれらのプロパティがすべて有効になります。JMS 仕様によれば、最初にプロパティをすべてクリアしない限り、アプリケーションでは、受信したメッセージのプロパティを変更できません。プロパティを変更せずに、受信メッセージを転送することができます。

注：アプリケーションが、XMSC\_WMQ\_MQMD\_READ\_ENABLED プロパティが true に設定された宛先からメッセージを受信して、XMSC\_WMQ\_MQMD\_WRITE\_ENABLED が true に設定された宛先にそのメッセージを転送すると、受信したメッセージの MQMD フィールド値はすべて、転送メッセージにコピーされます。プロパティの表

表 35. MQMD フィールドを表す Message オブジェクトのプロパティ		
プロパティ	説明	タイプ
JMS_IBM_MQMD_REPORT	レポート・メッセージのオプション	System.Int32
JMS_IBM_MQMD_MSGTYPE	メッセージ・タイプ	System.Int32
JMS_IBM_MQMD_EXPIRY	メッセージの存続時間	System.Int32
JMS_IBM_MQMD_FEEDBACK	フィードバックまたは理由コード	System.Int32
JMS_IBM_MQMD_ENCODING	メッセージ・データの数値エンコード	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	メッセージ・データの文字セット ID	System.Int32

表 35. MQMD フィールドを表す Message オブジェクトのプロパティ (続き)		
プロパティ	説明	タイプ
JMS_IBM_MQMD_FORMAT	メッセージ・データの形式名。	System.String
JMS_IBM_MQMD_PRIORITY 注: 0 から 9 までの範囲にない値を JMS_IBM_MQMD_PRIORITY に割り当てると、JMS 仕様違反になります。	メッセージ優先順位	System.Int32
JMS_IBM_MQMD_PERSISTENCE	メッセージの持続性	System.Int32
JMS_IBM_MQMD_MSGID 注: JMS 仕様には、メッセージ ID は JMS プロバイダーによって設定されている必要があり、かつ固有であるかヌルである必要があると記されています。 JMS_IBM_MQMD_MSGID に値を割り当てると、この値は JMSMessageID にコピーされます。つまり、値が JMS プロバイダーによって設定されず、固有の値でなくなる可能性があります。その場合は、JMS 仕様違反になります。	メッセージ ID	バイト配列 注: メッセージでバイト配列プロパティを使用すると、JMS 仕様違反します。
JMS_IBM_MQMD_CORRELID 注: スtring 「ID:」 で始まる JMS_IBM_MQMD_CORRELID に割り当てると、JMS 仕様違反になります。	相関 ID	バイト配列 注: メッセージでバイト配列プロパティを使用すると、JMS 仕様違反します。
JMS_IBM_MQMD_BACKOUTCOUNT	バックアウトのカウンター	System.Int32
JMS_IBM_MQMD_REPLYTOQ	応答キューの名前	System.String
JMS_IBM_MQMD_REPLYTOQMGR	応答キュー・マネージャーの名前	System.String
JMS_IBM_MQMD_USERIDENTIFIER	ユーザー ID	System.String
JMS_IBM_MQMD_ACCOUNTINGTOKEN	アカウントिंग・トークン	バイト配列 注: メッセージでバイト配列プロパティを使用すると、JMS 仕様違反します。
JMS_IBM_MQMD_APPLIDENTITYDATA	ID に関連するアプリケーション・データ	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	メッセージを書き込んだアプリケーションのタイプ	System.Int32
JMS_IBM_MQMD_PUTAPPLNAME	メッセージを書き込むアプリケーションの名前	System.String
JMS_IBM_MQMD_PUTDATE	メッセージを書き込んだ日付	System.String
JMS_IBM_MQMD_PUTTIME	メッセージを書き込んだ時刻	System.String
JMS_IBM_MQMD_APPLORIGINDATA	発生元に関するアプリケーション・データ	System.String

プロパティ	説明	タイプ
JMS_IBM_MQMD_GROUPID	グループ ID	バイト配列 注: メッセージでバイト配列プロパティを使用すると、JMS 仕様に違反します。
JMS_IBM_MQMD_MSGSEQNUMBER	グループ内のローカル・メッセージのシーケンス番号	System.Int32
JMS_IBM_MQMD_OFFSET	論理メッセージの先頭を起点とする、物理メッセージ中のデータのオフセット	System.Int32
JMS_IBM_MQMD_MSGFLAGS	メッセージ・フラグ	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	元のメッセージの長さ	System.Int32

詳しくは、[MQMD](#) を参照してください。

## 例

この例では、MQMD.UserIdentifier が「JoeBloggs」に設定されたキューまたはトピックにメッセージが書き込まれます。

```
// 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
// ...
```

JMS\_IBM\_MQMD\_USERIDENTIFIER を設定する前に、XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT を設定する必要があります。XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT の使用の詳細については、Message オブジェクト・プロパティを参照してください。

同様に、メッセージを受信する前に XMSC\_WMQ\_MQMD\_READ\_ENABLED を true に設定してから、メッセージの get メソッド (getStringProperty など) を使用することによって、MQMD フィールドの内容を抽出できます。受信するプロパティはすべて読み取り専用です。

この例では、メッセージの MQMD.AppIdentityData フィールドの値を保持する値フィールドがキューまたはトピックから取得されます。

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(XMSC_WMQ_MQMD_READ_ENABLED, XMSC_WMQ_MQMD_READ_ENABLED_YES);

// Receive a message
// ...
```

```
// Get desired MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

## MessageConsumer のプロパティ

MessageConsumer オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

プロパティ名	説明
<a href="#">XMSC_IS_SUBSCRIPTION_MULTICAST</a>	メッセージが WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されるかどうかを示します。このプロパティは読み取り専用です。
<a href="#">XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST</a>	メッセージが、信頼性の高いサービス品質を備えた WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されるかどうかを示します。このプロパティは読み取り専用です。

詳しくは、『[IMessageConsumer](#)』の『[.NET プロパティ](#)』を参照してください。

## MessageProducer のプロパティ

MessageProducer オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

詳しくは、[IMessageProducer](#) の [.NET プロパティ](#) を参照してください。

## Session のプロパティ

Session オブジェクトのプロパティの概要と、詳細な参照情報へのリンクを示します。

詳しくは、[ISession](#) の [.NET プロパティ](#) を参照してください。

## プロパティ定義

このトピック節では、各オブジェクト・プロパティの定義について説明します。

各プロパティ定義には、以下の情報が含まれます。

- プロパティのデータ・タイプ
- プロパティを持つオブジェクトの型
- Destination のプロパティの場合は、Uniform Resource Identifier (URI) で使用できる名前
- プロパティの詳細な説明
- プロパティの有効値
- プロパティのデフォルト値

名前が以下のいずれかのプレフィックスで始まるプロパティは、指定されたタイプの接続にのみ関連します。

### XMSC\_RTT

このプロパティは、ブローカーへのリアルタイム接続にのみ関連します。プロパティの名前は、名前付き定数としてヘッダー・ファイル `xmsc_rtt.h` に定義されます。

### XMSC\_WMQ

このプロパティは、アプリケーションが IBM WebSphere MQ キュー・マネージャーに接続している場合にのみ関連します。プロパティの名前は、名前付き定数としてヘッダー・ファイル `xmsc_wmq.h` に定義されます。

## XMSC\_WPM

このプロパティを使用するのは、アプリケーションが WebSphere サービス統合バスに接続する場合に限られます。プロパティの名前は、名前付き定数としてヘッダー・ファイル `xmsc_wpm.h` に定義されます。

プロパティの定義に特に明記されていない限り、残りのプロパティは、すべてのタイプの接続に関連します。プロパティの名前は、名前付き定数としてヘッダー・ファイル `xmsc.h` に定義されます。名前が `JMSX` というプレフィックスで始まるプロパティは、メッセージの JMS 定義のプロパティであり、名前が `JMS_IBM` というプレフィックスで始まるプロパティは、メッセージの IBM 定義のプロパティです。メッセージのプロパティの詳細については、73 ページの『[XMS メッセージのプロパティ](#)』を参照してください。

プロパティの定義に特に明記されていない限り、各プロパティは Point-to-Point と publish/subscribe の2つのドメインに関連します。

プロパティが読み取り専用と指定されていない限り、アプリケーションは任意のプロパティの値を取得して設定できます。

以下のプロパティが定義されています。

- [201 ページの『JMS\\_IBM\\_CHARACTER\\_SET』](#)
- [201 ページの『JMS\\_IBM\\_ENCODING』](#)
- [202 ページの『JMS\\_IBM\\_EXCEPTIONMESSAGE』](#)
- [202 ページの『JMS\\_IBM\\_EXCEPTIONPROBLEMDESTINATION』](#)
- [202 ページの『JMS\\_IBM\\_EXCEPTIONREASON』](#)
- [203 ページの『JMS\\_IBM\\_EXCEPTIONTIMESTAMP』](#)
- [203 ページの『JMS\\_IBM\\_FEEDBACK』](#)
- [203 ページの『JMS\\_IBM\\_FORMAT』](#)
- [203 ページの『JMS\\_IBM\\_LAST\\_MSG\\_IN\\_GROUP』](#)
- [204 ページの『JMS\\_IBM\\_MSGTYPE』](#)
- [204 ページの『JMS\\_IBM\\_PUTAPPLTYPE』](#)
- [204 ページの『JMS\\_IBM\\_PUTDATE』](#)
- [205 ページの『JMS\\_IBM\\_PUTTIME』](#)
- [205 ページの『JMS\\_IBM\\_REPORT\\_COA』](#)
- [205 ページの『JMS\\_IBM\\_REPORT\\_COD』](#)
- [206 ページの『JMS\\_IBM\\_REPORT\\_DISCARD\\_MSG』](#)
- [206 ページの『JMS\\_IBM\\_REPORT\\_EXCEPTION』](#)
- [207 ページの『JMS\\_IBM\\_REPORT\\_EXPIRATION』](#)
- [207 ページの『JMS\\_IBM\\_REPORT\\_NAN』](#)
- [208 ページの『JMS\\_IBM\\_REPORT\\_PAN』](#)
- [208 ページの『JMS\\_IBM\\_REPORT\\_PASS\\_CORREL\\_ID』](#)
- [208 ページの『JMS\\_IBM\\_REPORT\\_PASS\\_MSG\\_ID』](#)
- [209 ページの『JMS\\_IBM\\_SYSTEM\\_MESSAGEID』](#)
- [209 ページの『JMSX\\_APPID』](#)
- [210 ページの『JMSX\\_DELIVERY\\_COUNT』](#)
- [210 ページの『JMSX\\_GROUPID』](#)
- [210 ページの『JMSX\\_GROUPSEQ』](#)
- [210 ページの『JMSX\\_USERID』](#)
- [211 ページの『XMSC\\_CLIENT\\_ID』](#)
- [211 ページの『XMSC\\_CONNECTION\\_TYPE』](#)
- [211 ページの『XMSC\\_DELIVERY\\_MODE』](#)
- [212 ページの『XMSC\\_IC\\_PROVIDER\\_URL』](#)
- [212 ページの『XMSC\\_IC\\_SECURITY\\_AUTHENTICATION』](#)
- [213 ページの『XMSC\\_IC\\_SECURITY\\_CREDENTIALS』](#)
- [213 ページの『XMSC\\_IC\\_SECURITY\\_PRINCIPAL』](#)

[213 ページの『XMSC IC SECURITY PROTOCOL』](#)  
[213 ページの『XMSC IC URL』](#)  
[213 ページの『XMSC IS SUBSCRIPTION MULTICAST』](#)  
[214 ページの『XMSC IS SUBSCRIPTION RELIABLE MULTICAST』](#)  
[214 ページの『XMSC JMS MAJOR VERSION』](#)  
[214 ページの『XMSC JMS MINOR VERSION』](#)  
[214 ページの『XMSC JMS VERSION』](#)  
[214 ページの『XMSC MAJOR VERSION』](#)  
[214 ページの『XMSC MINOR VERSION』](#)  
[215 ページの『XMSC PASSWORD』](#)  
[215 ページの『XMSC PRIORITY』](#)  
[216 ページの『XMSC PROVIDER NAME』](#)  
[216 ページの『XMSC RTT BROKER PING INTERVAL』](#)  
[216 ページの『XMSC RTT CONNECTION PROTOCOL』](#)  
[216 ページの『XMSC RTT HOST NAME』](#)  
[216 ページの『XMSC RTT LOCAL ADDRESS』](#)  
[217 ページの『XMSC RTT MULTICAST』](#)  
[218 ページの『XMSC RTT PORT』](#)  
[218 ページの『XMSC TIME TO LIVE』](#)  
[218 ページの『XMSC USERID』](#)  
[219 ページの『XMSC VERSION』](#)  
[219 ページの『XMSC WMQ BROKER CONTROLQ』](#)  
[219 ページの『XMSC WMQ BROKER PUBQ』](#)  
[219 ページの『XMSC WMQ BROKER QMGR』](#)  
[220 ページの『XMSC WMQ BROKER SUBQ』](#)  
[220 ページの『XMSC WMQ BROKER VERSION』](#)  
[221 ページの『XMSC WMQ CCDURL』](#)  
[221 ページの『XMSC WMQ CCSID』](#)  
[221 ページの『XMSC WMQ CHANNEL』](#)  
[222 ページの『XMSC WMQ CONNECTION MODE』](#)  
[223 ページの『XMSC WMQ DUR SUBQ』](#)  
[223 ページの『XMSC WMQ ENCODING』](#)  
[224 ページの『XMSC WMQ FAIL IF QUIESCE』](#)  
[229 ページの『XMSC WMQ HOST NAME』](#)  
[229 ページの『XMSC WMQ LOCAL ADDRESS』](#)  
[230 ページの『XMSC WMQ MESSAGE SELECTION』](#)  
[230 ページの『XMSC WMQ MSG BATCH SIZE』](#)  
[230 ページの『XMSC WMQ POLLING INTERVAL』](#)  
[231 ページの『XMSC WMQ PORT』](#)  
[232 ページの『XMSC WMQ PUB ACK INTERVAL』](#)  
[232 ページの『XMSC WMQ QMGR CCSID』](#)  
[233 ページの『XMSC WMQ QUEUE MANAGER』](#)  
[233 ページの『XMSC WMQ RECEIVE EXIT』](#)  
[234 ページの『XMSC WMQ RECEIVE EXIT INIT』](#)  
[234 ページの『XMSC WMQ SECURITY EXIT』](#)  
[235 ページの『XMSC WMQ SECURITY EXIT INIT』](#)  
[235 ページの『XMSC WMQ SEND EXIT』](#)  
[235 ページの『XMSC WMQ SEND EXIT INIT』](#)  
[242 ページの『XMSC WMQ SYNCPOINT ALL GETS』](#)  
[243 ページの『XMSC WMQ TARGET CLIENT』](#)  
[243 ページの『XMSC WMQ TEMP Q PREFIX』](#)

[244 ページの『XMSC\\_WMQ\\_TEMPORARY\\_MODEL』](#)  
[244 ページの『XMSC\\_WPM\\_BUS\\_NAME』](#)  
[245 ページの『XMSC\\_WPM\\_CONNECTION\\_PROTOCOL』](#)  
[245 ページの『XMSC\\_WPM\\_CONNECTION\\_PROXIMITY』](#)  
[245 ページの『XMSC\\_WPM\\_DUR\\_SUB\\_HOME』](#)  
[246 ページの『XMSC\\_WPM\\_HOST\\_NAME』](#)  
[246 ページの『XMSC\\_WPM\\_LOCAL\\_ADDRESS』](#)  
[247 ページの『XMSC\\_WPM\\_ME\\_NAME』](#)  
[247 ページの『XMSC\\_WPM\\_NON\\_PERSISTENT\\_MAP』](#)  
[247 ページの『XMSC\\_WPM\\_PERSISTENT\\_MAP』](#)  
[248 ページの『XMSC\\_WPM\\_PORT』](#)  
[248 ページの『XMSC\\_WPM\\_PROVIDER\\_ENDPOINTS』](#)  
[249 ページの『XMSC\\_WPM\\_TARGET\\_GROUP』](#)  
[249 ページの『XMSC\\_WPM\\_TARGET\\_SIGNIFICANCE』](#)  
[249 ページの『XMSC\\_WPM\\_TARGET\\_TRANSPORT\\_CHAIN』](#)  
[250 ページの『XMSC\\_WPM\\_TARGET\\_TYPE』](#)  
[250 ページの『XMSC\\_WPM\\_TEMP\\_Q\\_PREFIX』](#)  
[251 ページの『XMSC\\_WPM\\_TEMP\\_TOPIC\\_PREFIX』](#)  
[251 ページの『XMSC\\_WPM\\_TOPIC\\_SPACE』](#)

## **JMS\_IBM\_CHARACTER\_SET**

**データ型:**

System.Int32

**プロパティ:**

メッセージ

XMS クライアントがメッセージを目的の宛先に転送する場合、メッセージの本文中の文字データ・ストリングが設定されるコード化文字セットの ID (CCSID)、またはコード・ページ。XMS では、このプロパティには数値が指定され、CCSID にマップされます。ただし、このプロパティは JMS プロパティに基づいているため、ストリング型の値を持ち、この数値 CCSID を表す Java 文字セットにマップされます。このプロパティは、[XMSC\\_WMQ\\_CCSID](#) プロパティによって宛先に指定されたすべての CCSID に優先します。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

## **JMS\_IBM\_ENCODING**

**データ型:**

System.Int32

**プロパティ:**

メッセージ

XMS クライアントがメッセージを目的の宛先に転送する場合、メッセージの本文中の数値データをどのように表記するかを表します。このプロパティは、[XMSC\\_WMQ\\_ENCODING](#) プロパティによって宛先に指定されたすべてのエンコード方式に優先します。このプロパティは、2 進整数、パック 10 進数、および浮動小数点数の表記を指定します。

このプロパティの有効値は、メッセージ記述子の *Encoding* フィールドに指定できる値と同じです。*Encoding* フィールドの詳細については、「[IBM WebSphere MQ アプリケーション・プログラミング・リファレンス](#)」を参照してください。

アプリケーションは、以下の名前付き定数を使用してプロパティを設定できます。

**名前付き定数**

**意味**

MQENC\_INTEGER\_NORMAL

標準の整数エンコード方式

名前付き定数	意味
MQENC_INTEGER_REVERSED	逆方向の整数エンコード方式
MQENC_DECIMAL_NORMAL	標準のパック 10 進エンコード方式
MQENC_DECIMAL_REVERSED	逆方向のパック 10 進エンコード方式
MQENC_FLOAT_IEEE_NORMAL	標準の IEEE 浮動小数点エンコード方式
MQENC_FLOAT_IEEE_REVERSED	逆方向の IEEE 浮動小数点エンコード方式
MQENC_FLOAT_S390	z/OS® アーキテクチャー浮動小数点エンコード
MQENC_NATIVE	ネイティブのマシン・エンコード方式

プロパティの値を設定するために、アプリケーションは、以下に示す 3 つの定数を加算できます。

- 2 進整数の表記を指定するための、名前が MQENC\_INTEGER で始まる定数
- パック 10 進整数の表記を指定するための、名前が MQENC\_DECIMAL で始まる定数
- 浮動小数点数の表記を指定するための、名前が MQENC\_FLOAT で始まる定数

あるいは、アプリケーションは、その値が環境に依存する MQENC\_NATIVE にプロパティを設定できます。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

### **JMS\_IBM\_EXCEPTIONMESSAGE**

**データ型:**  
 スtring

**プロパティ:**  
 メッセージ

メッセージが例外の宛先に送信された理由を説明するテキストです。このプロパティは読み取り専用です。

このプロパティが関連するのは、アプリケーションがサービス統合バスに接続し、例外の宛先からメッセージを受信した場合に限られます。

### **JMS\_IBM\_EXCEPTIONPROBLEMDESTINATION**

**データ型:**  
 スtring

**プロパティ:**  
 メッセージ

メッセージが例外の宛先に送信される前に、そのメッセージが存在した宛先の名前です。

このプロパティが関連するのは、アプリケーションがサービス統合バスに接続し、例外の宛先からメッセージを受信した場合に限られます。

### **JMS\_IBM\_EXCEPTIONREASON**

**データ型:**  
 System.Int32

**プロパティ:**  
 メッセージ

メッセージが例外の宛先に送信された理由を示す理由コードです。

可能性のあるすべての理由コードのリストについては、WebSphere ApplicationServer に付属の Javadoc ツールによって生成される文書内にある `com.ibm.websphere.sib.SIRCConstants` クラスの定義を参照してください。

このプロパティが関連するのは、アプリケーションがサービス統合バスに接続し、例外の宛先からメッセージを受信した場合に限られます。

### **JMS\_IBM\_EXCEPTIONTIMESTAMP**

**データ型:**  
System.Int64

**プロパティ:**  
メッセージ

メッセージが例外の宛先に送信された時刻です。

この時刻は、1970年1月1日 00:00:00 GMT からの経過時間をミリ秒単位で表現したものです。

このプロパティが関連するのは、アプリケーションがサービス統合バスに接続し、例外の宛先からメッセージを受信した場合に限られます。

### **JMS\_IBM\_FEEDBACK**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

レポート・メッセージの種類を示すコードです。

このプロパティの有効値は、メッセージ記述子の **Feedback** フィールドに指定できるフィードバック・コードおよび理由コードです。**Feedback** フィールドの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

デフォルトでは、このプロパティは設定されていません。

### **JMS\_IBM\_FORMAT**

**データ型:**  
ストリング

**プロパティ:**  
メッセージ

メッセージ内にあるアプリケーション・データの種類の種類です。

このプロパティの有効値は、メッセージ記述子の **Format** フィールドに指定できる値と同じです。**Format** フィールドの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

### **JMS\_IBM\_LAST\_MSG\_IN\_GROUP**

**データ型:**  
System.Boolean

**プロパティ:**  
メッセージ

メッセージがメッセージ・グループ内の最後のメッセージであるかどうかを示します。

メッセージがメッセージ・グループ内の最後のメッセージである場合は、このプロパティを **true** に設定します。それ以外の場合は、このプロパティを **false** に設定するか、このプロパティを設定しないようにします。デフォルトでは、このプロパティは設定されていません。

値 **true** は、状況フラグ **MQMF\_LAST\_MSG\_IN\_GROUP** に対応し、このフラグはメッセージ記述子の **MsgFlags** フィールドで指定できます。このフラグの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

このプロパティは **publish/subscribe** ドメイン内では無視され、アプリケーションがサービス統合バスに接続している場合は関係ありません。

## **JMS\_IBM\_MSGTYPE**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

メッセージのタイプ。

プロパティの有効値は以下のとおりです。

有効値	意味
MQMT_DATAGRAM	メッセージは、応答が不要なメッセージです。
MQMT_REQUEST	メッセージは、応答が必要なメッセージです。
MQMT_REPLY	このメッセージは、応答メッセージです。
MQMT_REPORT	このメッセージは、レポート・メッセージです。

これらの値は、メッセージ記述子の **MsgType** フィールドに指定できるメッセージのタイプに対応します。**MsgType** フィールドの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

## **JMS\_IBM\_PUTAPPLTYPE**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

メッセージを送信したアプリケーションのタイプです。

このプロパティの有効値は、メッセージ記述子の **PutAppType** フィールドに指定できるアプリケーション・タイプです。**PutAppType** フィールドの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

## **JMS\_IBM\_PUTDATE**

**データ型:**  
string

**プロパティ:**  
メッセージ

メッセージが送信された日付です。

このプロパティの有効値は、メッセージ記述子の **PutDate** フィールドに指定できる値と同じです。**PutDate** フィールドの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

### **JMS\_IBM\_PUTTIME**

**データ型:**  
     ストリング

**プロパティ:**  
     メッセージ

メッセージが送信された時刻です。

このプロパティの有効値は、メッセージ記述子の **PutTime** フィールドに指定できる値と同じです。**PutTime** フィールドの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがサービス統合バスに接続している場合、このプロパティは関係ありません。

### **JMS\_IBM\_REPORT\_COA**

**データ型:**  
     System.Int32

**プロパティ:**  
     メッセージ

「到着時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。

プロパティの有効値は以下のとおりです。

<b>有効値</b>	<b>意味</b>
MQRO_COA	「到着時の確認」レポート・メッセージを要求します。元のメッセージからのアプリケーション・データはレポート・メッセージに組み込みません。
MQRO_COA_WITH_DATA	「到着時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データのうち先頭の 100 バイトをレポート・メッセージに組み込みます。
MQRO_COA_WITH_FULL_DATA	「到着時の確認」レポート・メッセージを要求し、元のメッセージからのすべてのアプリケーション・データをレポート・メッセージに組み込みます。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

### **JMS\_IBM\_REPORT\_COD**

**データ型:**  
     System.Int32

**プロパティ:**  
     メッセージ

「配信時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。

プロパティの有効値は以下のとおりです。

有効値	意味
MQRO_COD	「配信時の確認」レポート・メッセージを要求します。元のメッセージからのアプリケーション・データはレポート・メッセージに組み込みません。
MQRO_COD_WITH_DATA	「配信時の確認」レポート・メッセージを要求し、元のメッセージからのアプリケーション・データのうち先頭の 100 バイトをレポート・メッセージに組み込みます。
MQRO_COD_WITH_FULL_DATA	「配信時の確認」レポート・メッセージを要求し、元のメッセージからのすべてのアプリケーション・データをレポート・メッセージに組み込みます。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

デフォルトでは、このプロパティは設定されていません。

## **JMS\_IBM\_REPORT\_DISCARD\_MSG**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

メッセージを目的の宛先に配信できない場合に、そのメッセージを廃棄することを要求します。

メッセージを目的の宛先に配信できない場合に、そのメッセージを廃棄するには、このプロパティを `MQRO_DISCARD_MSG` に設定します。そうではなく、メッセージを送達不能キューに書き込むか、例外の宛先に送信することを要求する場合は、このプロパティを設定しないでください。デフォルトでは、このプロパティは設定されていません。

`MQRO_DISCARD_MSG` という値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応します。このオプションの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

## **JMS\_IBM\_REPORT\_EXCEPTION**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

例外レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。

プロパティの有効値は以下のとおりです。

有効値	意味
MQRO_EXCEPTION	例外レポート・メッセージを要求します。元のメッセージからのアプリケーション・データはレポート・メッセージに組み込みません。

## 有効値

MQRO\_EXCEPTION\_WITH\_DATA

## 意味

例外レポート・メッセージを要求し、元のメッセージからのアプリケーション・データのうち先頭の 100 バイトをレポート・メッセージに組み込みます。

MQRO\_EXCEPTION\_WITH\_FULL\_DATA

例外レポート・メッセージを要求し、元のメッセージからのすべてのアプリケーション・データをレポート・メッセージに組み込みます。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

デフォルトでは、このプロパティは設定されていません。

## JMS\_IBM\_REPORT\_EXPIRATION

### データ型:

System.Int32

### プロパティ:

メッセージ

期限満了レポート・メッセージを要求し、元のメッセージからのアプリケーション・データをレポート・メッセージにどの程度組み込む必要があるかを指定します。

プロパティの有効値は以下のとおりです。

## 有効値

MQRO\_EXPIRATION

## 意味

期限満了レポート・メッセージを要求します。元のメッセージからのアプリケーション・データはレポート・メッセージに組み込みません。

MQRO\_EXPIRATION\_WITH\_DATA

期限満了レポート・メッセージを要求し、元のメッセージからのアプリケーション・データのうち先頭の 100 バイトをレポート・メッセージに組み込みます。

MQRO\_EXPIRATION\_WITH\_FULL\_DATA

期限満了レポート・メッセージを要求し、元のメッセージからのすべてのアプリケーション・データをレポート・メッセージに組み込みます。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ* アプリケーション・プログラミング・リファレンス」を参照してください。

デフォルトでは、このプロパティは設定されていません。

## JMS\_IBM\_REPORT\_NAN

### データ型:

System.Int32

### プロパティ:

メッセージ

否定アクション通知レポート・メッセージを要求します。

否定アクション通知レポート・メッセージを要求するには、このプロパティを MQRO\_NAN に設定します。否定アクション通知レポート・メッセージが必要でない場合は、このプロパティを設定しないでください。デフォルトでは、このプロパティは設定されていません。

MQRO\_NAN という値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応します。このオプションの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

### **JMS\_IBM\_REPORT\_PAN**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

肯定アクション通知レポート・メッセージを要求します。

肯定アクション通知レポート・メッセージを要求するには、このプロパティを MQRO\_PAN に設定します。肯定アクション通知レポート・メッセージが必要でない場合は、このプロパティを設定しないでください。デフォルトでは、このプロパティは設定されていません。

MQRO\_PAN という値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応します。このオプションの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

### **JMS\_IBM\_REPORT\_PASS\_CORREL\_ID**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

任意のレポート・メッセージまたは応答メッセージの相関 ID を元のメッセージの相関 ID と同じにするという要求です。

プロパティの有効値は以下のとおりです。

有効値	意味
MQRO_PASS_CORREL_ID	任意のレポート・メッセージまたは応答メッセージの相関 ID を元のメッセージの相関 ID と同じにするという要求です。
MQRO_COPY_MSG_ID_TO_CORREL_ID	任意のレポート・メッセージまたは応答メッセージの相関 ID を元のメッセージのメッセージ ID と同じにするという要求です。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

このプロパティのデフォルト値は MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID です。

### **JMS\_IBM\_REPORT\_PASS\_MSG\_ID**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

任意のレポート・メッセージまたは応答メッセージのメッセージ ID を元のメッセージのメッセージ ID と同じにするという要求です。

プロパティの有効値は以下のとおりです。

有効値	意味
MQRO_PASS_MSG_ID	任意のレポート・メッセージまたは応答メッセージのメッセージ ID を元のメッセージのメッセージ ID と同じにするという要求です。
MQRO_NEW_MSG_ID	レポート・メッセージまたは応答メッセージごとに新しいメッセージ ID を生成するという要求です。

これらの値は、メッセージ記述子の **Report** フィールドに指定できるレポート・オプションに対応しています。これらのオプションの詳細については、「*IBM WebSphere MQ アプリケーション・プログラミング・リファレンス*」を参照してください。

このプロパティのデフォルト値は MQRO\_NEW\_MSG\_ID です。

## **JMS\_IBM\_RETAIN**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

このプロパティを設定すると、キュー・マネージャーは、メッセージを保存パブリケーションとして扱うように指定されます。サブスクライバーは、トピックからメッセージを受け取る時に、前のリリースで受け取っていたメッセージのほかに、サブスクライブ直後に追加のメッセージを受け取ることがあります。それらのメッセージは、サブスクライブしたトピックのオプションの保存パブリケーションです。サブスクリプションに合致するトピックごとに、保存パブリケーションがある場合は、そのパブリケーションも、サブスクライブしているメッセージ・コンシューマーに配信できるようになります。

RETAIN\_PUBLICATION は、このプロパティで唯一有効な値です。デフォルトでは、このプロパティは設定されていません。

注: このプロパティが関連するのは、パブリッシュ/サブスクライブ・ドメインの場合に限られます。

## **JMS\_IBM\_SYSTEM\_MESSAGEID**

**データ型:**  
ストリング

**プロパティ:**  
メッセージ

サービス統合バスの内部でメッセージを一意的に識別する ID です。このプロパティは読み取り専用です。

このプロパティが関連するのは、アプリケーションがサービス統合バスに接続している場合に限られます。

## **JMSX\_APPID**

**データ型:**  
ストリング

**プロパティ:**  
メッセージ

メッセージを送信したアプリケーションの名前です。

このプロパティは JMS 定義のプロパティで、JMSXAppID という JMS 名が付いています。このプロパティの詳細については、「*Java Message Service Specification, Version 1.1*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

## **JMSX\_DELIVERY\_COUNT**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

メッセージ配信の試行回数です。

このプロパティは JMS 定義のプロパティで、JMSXDeliveryCount という JMS 名が付いています。このプロパティの詳細については、「*Java Message Service Specification, Version 1.1*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

## **JMSX\_GROUPID**

**データ型:**  
ストリング

**プロパティ:**  
メッセージ

メッセージが属するメッセージ・グループの ID です。

このプロパティは JMS 定義のプロパティで、JMSXGroupID という JMS 名が付いています。このプロパティの詳細については、「*Java Message Service Specification, Version 1.1*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

## **JMSX\_GROUPSEQ**

**データ型:**  
System.Int32

**プロパティ:**  
メッセージ

メッセージ・グループ内にあるメッセージのシーケンス番号です。

このプロパティは JMS 定義のプロパティで、JMSXGroupSeq という JMS 名が付いています。このプロパティの詳細については、「*Java Message Service Specification, Version 1.1*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

## **JMSX\_USERID**

**データ型:**  
ストリング

**プロパティ:**  
メッセージ

メッセージを送信したアプリケーションに関連付けられているユーザー ID です。

このプロパティは JMS 定義のプロパティで、JMSXUserID という JMS 名が付いています。このプロパティの詳細については、「*Java Message Service Specification, Version 1.1*」を参照してください。

デフォルトでは、このプロパティは設定されていません。

ブローカーへのリアルタイム接続の場合、このプロパティは無効です。

## **XMSC\_ASYNC\_EXCEPTIONS**

データ型:

System.Int32

プロパティ:

ConnectionFactory

このプロパティは、XMS が ExceptionListener への通知を、接続が切断されたときのみ行うか、または XMS API 呼び出しに対して非同期に例外が発生したときに行うかを決定します。このプロパティは、ExceptionListener が登録されているこの ConnectionFactory から作成されたすべての接続に適用されます。

このプロパティの有効な値は以下のとおりです。

### **XMSC\_ASYNC\_EXCEPTIONS\_ALL**

同期 API 呼び出しの有効範囲外で非同期に検出されたすべての例外、および接続切断の例外はすべて ExceptionListener に送信されます。

### **XMSC\_ASYNC\_EXCEPTIONS\_CONNECTIONBROKEN**

切断された接続を示す例外のみが ExceptionListener に送信されます。非同期処理中に起きる他のすべての例外は ExceptionListener には報告されず、そのためアプリケーションにはそれらの例外は通知されません。

デフォルトでは、このプロパティは XMSC\_ASYNC\_EXCEPTIONS\_ALL に設定されます。

## **XMSC\_CLIENT\_ID**

データ型:

ストリング

プロパティ:

ConnectionFactory

接続のクライアント ID です。

クライアント ID は、publish/subscribe ・ドメイン内の永続サブスクリプションをサポートするためにのみ使用され、Point-to-Point ドメインでは無視されます。クライアント ID の設定についての詳細は、[24 ページの『ConnectionFactory オブジェクトと Connection オブジェクト』](#)を参照してください。

このプロパティは、ブローカーへのリアルタイム接続には関連していません。

## **XMSC\_CONNECTION\_TYPE**

データ型:

System.Int32

プロパティ:

ConnectionFactory

アプリケーションの接続先となるメッセージング・サーバーのタイプです。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_CT_RTT	ブローカーへのリアルタイム接続です。
XMSC_CT_WMQ	IBM WebSphere MQ キュー・マネージャーへの接続です。
XMSC_CT_WPM	WebSphere サービス統合バスへの接続です。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_DELIVERY\_MODE**

データ型:

System.Int32

**プロパティ:**  
Destination

**URI で使用される名前:**

persistence (IBM WebSphere MQ 宛先の場合)  
deliveryMode (WebSphere デフォルト・メッセージング・プロバイダーの宛先の場合)

宛先に送信されたメッセージの送達モードです。

プロパティの有効値は以下のとおりです。

**有効値**

XMSC\_DELIVERY\_NOT\_PERSISTENT

**意味**

宛先に送信されたメッセージは、非永続です。メッセージ・プロデューサーのデフォルトの送達モード、または Send 呼び出しに指定されている任意の送達モードは無視されます。宛先が IBM WebSphere MQ キューである場合は、キュー属性 *DefPersistence* の値も無視されます。

XMSC\_DELIVERY\_PERSISTENT

宛先に送信されたメッセージは、永続です。メッセージ・プロデューサーのデフォルトの送達モード、または Send 呼び出しに指定されている任意の送達モードは無視されます。宛先が IBM WebSphere MQ キューである場合は、キュー属性 *DefPersistence* の値も無視されます。

XMSC\_DELIVERY\_AS\_APP

宛先に送信されたメッセージの送達モードは、Send 呼び出しに指定されている送達モードです。Send 呼び出しに送達モードが指定されていない場合は、代わりにメッセージ・プロデューサーのデフォルトの送達モードが使用されます。宛先が IBM WebSphere MQ キューである場合は、キュー属性 *DefPersistence* の値も無視されます。

XMSC\_DELIVERY\_AS\_DEST

宛先が IBM WebSphere MQ キューである場合、キューに書き込まれたメッセージの送達モードは、キュー属性 *DefPersistence* の値で指定された送達モードになります。メッセージ・プロデューサーのデフォルトの送達モード、または Send 呼び出しに指定されている任意の送達モードは無視されます。

宛先が IBM WebSphere MQ キューではない場合、この値の意味は XMSC\_DELIVERY\_AS\_APP の意味と同じです。

デフォルト値は XMSC\_DELIVERY\_AS\_APP です。

**XMSC\_IC\_PROVIDER\_URL**

**データ型:**  
string

**プロパティ:**  
InitialContext

JNDI ネーミング・ディレクトリーの位置を指定するために使用します。その位置を指定すれば、COS ネーミング・サービスが Web サービスと同じサーバーに存在する必要はなくなります。

**XMSC\_IC\_SECURITY\_AUTHENTICATION**

**データ型:**  
string

**プロパティ:**  
InitialContext

Java コンテキスト・インターフェース SECURITY\_AUTHENTICATION に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。

### ***XMSC\_IC\_SECURITY\_CREDENTIALS***

**データ型:**  
ストリング

**プロパティ:**  
InitialContext

Java コンテキスト・インターフェース SECURITY\_CREDENTIALS に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。

### ***XMSC\_IC\_SECURITY\_PRINCIPAL***

**データ型:**  
ストリング

**プロパティ:**  
InitialContext

Java コンテキスト・インターフェース SECURITY\_PRINCIPAL に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。

### ***XMSC\_IC\_SECURITY\_PROTOCOL***

**データ型:**  
ストリング

**プロパティ:**  
InitialContext

Java コンテキスト・インターフェース SECURITY\_PROTOCOL に基づいています。このプロパティは COS ネーミング・コンテキストにのみ適用されます。

### ***XMSC\_IC\_URL***

**データ型:**  
ストリング

**プロパティ:**  
InitialContext

LDAP コンテキストや FileSystem コンテキストの場合は、管理対象オブジェクトを収容しているリポジトリーのアドレスです。

COS ネーミング・コンテキストの場合は、ディレクトリー内のオブジェクトを検索する Web サービスのアドレスです。

### ***XMSC\_IS\_SUBSCRIPTION\_MULTICAST***

**データ型:**  
System.Boolean

**プロパティ:**  
MessageConsumer

メッセージが WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されるかどうかを示します。このプロパティは読み取り専用です。

メッセージが WebSphere MQ Multicast Transport を使用してメッセージ・コンシューマーに配信される場合、このプロパティの値は true です。それ以外の場合、値は false です。

このプロパティは、ブローカーへのリアルタイム接続にのみ関連します。

## ***XMSC\_IS\_SUBSCRIPTION\_RELIABLE\_MULTICAST***

**データ型:**

System.Boolean

**プロパティ:**

MessageConsumer

メッセージが、信頼性の高いサービス品質を備えた WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されるかどうかを示します。このプロパティは読み取り専用です。

メッセージが、信頼性の高いサービス品質を備えた WebSphere MQ Multicast Transport を使用してメッセージ・コンシューマーに配信される場合、このプロパティの値は true です。それ以外の場合、値は false です。

このプロパティは、ブローカーへのリアルタイム接続にのみ関連します。

## ***XMSC\_JMS\_MAJOR\_VERSION***

**データ型:**

System.Int32

**プロパティ:**

ConnectionMetaData

XMS が準拠する JMS 仕様のメジャー・バージョン番号です。このプロパティは読み取り専用です。

## ***XMSC\_JMS\_MINOR\_VERSION***

**データ型:**

System.Int32

**プロパティ:**

ConnectionMetaData

XMS が準拠する JMS 仕様のマイナー・バージョン番号です。このプロパティは読み取り専用です。

## ***XMSC\_JMS\_VERSION***

**データ型:**

ストリング

**プロパティ:**

ConnectionMetaData

XMS が準拠する JMS 仕様のバージョン ID です。このプロパティは読み取り専用です。

## ***XMSC\_MAJOR\_VERSION***

**データ型:**

System.Int32

**プロパティ:**

ConnectionMetaData

XMS クライアントのバージョン番号です。このプロパティは読み取り専用です。

## ***XMSC\_MINOR\_VERSION***

**データ型:**

System.Int32

**プロパティ:**

ConnectionMetaData

XMS クライアントのリリース番号です。このプロパティは読み取り専用です。

## **XMSC\_PASSWORD**

**データ型:**  
バイト配列

**プロパティ:**  
ConnectionFactory

アプリケーションがメッセージング・サーバーに接続しようとしているときに、このアプリケーションを認証するために使用するパスワードです。このパスワードは、XMSC\_USERID プロパティと一緒に使用します。

デフォルトでは、このプロパティは設定されていません。

分散プラットフォーム上の IBM WebSphere MQ に接続する場合、接続ファクトリーの XMSC\_USERID プロパティを設定するときは、ログオン・ユーザーの **userid** と一致している必要があります。これらのプロパティを設定しない場合、キュー・マネージャーはデフォルトでログオン・ユーザーの **userid** を使用します。個々のユーザーの接続レベル認証がさらに必要な場合には、IBM WebSphere MQ で構成済みのクライアント認証出口を作成できます。クライアント認証出口の作成に関する詳細は、IBM WebSphere MQ Clients のマニュアルの「認証」のトピックでさらに調べることができます。

IBM WebSphere MQ on z/OS に接続するときにユーザーを認証するには、セキュリティー出口を使用する必要があります。

## **XMSC\_PRIORITY**

**データ型:**  
System.Int32

**プロパティ:**  
Destination

**URI で使用される名前:**  
priority

宛先に送信されたメッセージの優先順位です。

プロパティの有効値は以下のとおりです。

<b>有効値</b>	<b>意味</b>
0 (最低の優先順位) から 9 (最高の優先順位) までの範囲の整数	宛先に送信されたメッセージには、指定された優先順位があります。メッセージ・プロデューサーのデフォルトの優先順位、および Send 呼び出しに指定されているすべての優先順位は無視されます。宛先が IBM WebSphere MQ キューである場合は、キュー属性の値 <b>DefPriority</b> も無視されます。
XMSC_PRIORITY_AS_APP	宛先に送信されたメッセージの優先順位は、Send 呼び出しに指定されている優先順位です。Send 呼び出しに優先順位が指定されていない場合は、代わりにメッセージ・プロデューサーのデフォルトの優先順位が使用されます。宛先が IBM WebSphere MQ キューである場合は、キュー属性の値 <b>DefPriority</b> も無視されません。
XMSC_PRIORITY_AS_DEST	宛先が IBM WebSphere MQ キューである場合、キューに書き込まれたメッセージの優先順位は、キュー属性 <b>DefPriority</b> の値で指定された優先順位になります。メッセージ・プロデューサーのデフォルトの優先順位、および Send 呼び出しに指定されているすべての優先順位は無視されます。  宛先が IBM WebSphere MQ キューではない場合、この値の意味は XMSC_PRIORITY_AS_APP の意味と同じです。

デフォルト値は XMSC\_PRIORITY\_AS\_APP です。

WebSphere MQ Real-Time Transport および WebSphere MQ Multicast Transport には、メッセージの優先順位に基づく動作はありません。

### ***XMSC\_PROVIDER\_NAME***

データ型:  
    ストリング

プロパティ:  
    ConnectionMetaData

XMS クライアントのプロバイダーです。このプロパティは読み取り専用です。

### ***XMSC\_RTT\_BROKER\_PING\_INTERVAL***

データ型:  
    System.Int32

プロパティ:  
    ConnectionFactory

XMS .NET がリアルタイム・メッセージング・サーバーへの接続を検査することでアクティビティの検出を行うまでの時間間隔(ミリ秒単位)。アクティビティが何も検出されないと、クライアントは ping を開始します。ping に対する応答が検出されないと、接続は閉じられます。

このプロパティのデフォルト値は 30000 です。

### ***XMSC\_RTT\_CONNECTION\_PROTOCOL***

データ型:  
    System.Int32

プロパティ:  
    ConnectionFactory

ブローカーへのリアルタイム接続に使用される通信プロトコルです。

このプロパティの値は、TCP/IP を使用したブローカーへのリアルタイム接続という意味の XMSC\_RTT\_CP\_TCP にする必要があります。デフォルト値は XMSC\_RTT\_CP\_TCP です。

### ***XMSC\_RTT\_HOST\_NAME***

データ型:  
    ストリング

プロパティ:  
    ConnectionFactory

ブローカーを実行するシステムのホスト名または IP アドレスです。

このプロパティは、ブローカーを識別するために、[XMSC\\_RTT\\_PORT](#) プロパティと一緒に使用します。

デフォルトでは、このプロパティは設定されていません。

### ***XMSC\_RTT\_LOCAL\_ADDRESS***

データ型:  
    ストリング

プロパティ:  
    ConnectionFactory

ブローカーへのリアルタイム接続に使用するローカル・ネットワーク・インターフェースのホスト名または IP アドレスです。

このプロパティが有用なのは、アプリケーションを実行しているシステムに複数のネットワーク・インターフェースがあり、リアルタイム接続にどのインターフェースを使用する必要があるかを指定できることが必要な場合に限ります。システムが備えるネットワーク・インターフェースが 1 つのみである場合、

使用できるのはそのインターフェースのみです。システムに複数のネットワーク・インターフェースがあり、このプロパティを設定していない場合、ネットワーク・インターフェースはランダムに選択されます。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_RTT\_MULTICAST**

### **データ型:**

System.Int32

### **プロパティ:**

ConnectionFactory および Destination

### **URI で使用される名前:**

multicast

接続ファクトリーまたは宛先のマルチキャスト設定です。このプロパティを設定できるのは、内容がトピックである宛先のみです。

アプリケーションがこのプロパティを使用する目的は、以下の2つです。1つは、ブローカーへのリアルタイム接続と関連してマルチキャストを使用可能にすることで、もう1つは、マルチキャストが使用可能になった場合に、ブローカーからメッセージ・コンシューマーにメッセージを配信するときにマルチキャストを使用する方法を正確に指定することです。このプロパティは、メッセージ・プロデューサーによるブローカーへのメッセージ送信方法については影響しません。

プロパティの有効値は以下のとおりです。

### **有効値**

XMSC\_RTT\_MULTICAST\_DISABLED

XMSC\_RTT\_MULTICAST\_ASCF

XMSC\_RTT\_MULTICAST\_ENABLED

XMSC\_RTT\_MULTICAST\_RELIABLE

### **意味**

メッセージは WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されません。この値は、ConnectionFactory オブジェクトのデフォルト値です。

メッセージは、メッセージ・コンシューマーに関連付けられた接続ファクトリーのマルチキャスト設定に応じて、メッセージ・コンシューマーに送達されます。接続ファクトリーのマルチキャスト設定は、接続経路を作成するときに指定します。この値は Destination オブジェクトに対してのみ有効であり、Destination オブジェクトのデフォルト値になります。

ブローカーでのマルチキャストに合わせてトピックを構成した場合、メッセージは WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されます。信頼性の高いマルチキャストに合わせてトピックを構成した場合は、信頼性の高いサービス品質が使用されません。

ブローカーでの信頼性の高いマルチキャストに合わせてトピックを構成した場合、メッセージは、信頼性の高いサービス品質を備えた WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されます。信頼性の高いマルチキャストに合わせてトピックを構成しなかった場合は、このトピックに対してメッセージ・コンシューマーを作成することはできません。

## 有効値

XMSC\_RTT\_MULTICAST\_NOT\_RELIABLE

## 意味

ブローカーでのマルチキャストに合わせてトピックを構成した場合、メッセージは WebSphere MQ Multicast Transport によってメッセージ・コンシューマーに配信されます。信頼性の高いマルチキャストに合わせてトピックを構成した場合でも、信頼性の高いサービス品質は使用されません。

## **XMSC\_RTT\_PORT**

### データ型:

System.Int32

### プロパティ:

ConnectionFactory

ブローカーが着信要求を listen するポートの数です。ブローカーでは、Real-timeInput または Real-timeOptimizedFlow メッセージ処理ノードを構成して、このポートを listen する必要があります。

このプロパティは、ブローカーを識別するために、XMSC\_RTT\_HOST\_NAME プロパティと一緒に使用します。

このプロパティのデフォルト値は XMSC\_RTT\_DEFAULT\_PORT、つまり 1506 です。

## **XMSC\_TIME\_TO\_LIVE**

### データ型:

System.Int32

### プロパティ:

Destination

### URI で使用される名前:

expiry (IBM WebSphere MQ 宛先の場合)

timeToLive (WebSphere デフォルト・メッセージング・プロバイダーの宛先の場合)

宛先に送信されたメッセージの存続時間です。

プロパティの有効値は以下のとおりです。

## 有効値

0

正整数

XMSC\_TIME\_TO\_LIVE\_AS\_APP

## 意味

宛先に送信されたメッセージには有効期限がありません。

宛先に送信されたメッセージには、指定された存続時間(ミリ秒単位)があります。メッセージ・プロデューサーのデフォルトの存続時間、または Send 呼び出しに指定されているすべての存続時間は無視されます。

宛先に送信されたメッセージの存続時間は、Send 呼び出しに指定されている存続時間です。Send 呼び出しに存続時間が指定されていない場合は、代わりにメッセージ・プロデューサーのデフォルトの存続時間が使用されます。

デフォルト値は XMSC\_TIME\_TO\_LIVE\_AS\_APP です。

## **XMSC\_USERID**

### データ型:

ストリング

### プロパティ:

ConnectionFactory

アプリケーションがメッセージング・サーバーに接続しようとしているときに、このアプリケーションを認証するために使用するユーザー ID です。このユーザー ID は、XMSC\_PASSWORD プロパティと一緒に使われます。

デフォルトでは、このプロパティは設定されていません。

IBM WebSphere MQ 分散プラットフォームに接続する場合、接続ファクトリーの `XMSC_USERID` プロパティを設定するときは、ログオン・ユーザーの **userid** と一致する必要があります。これらのプロパティを設定しない場合、キュー・マネージャーはデフォルトでログオン・ユーザーの **userid** を使います。個々のユーザーの追加の接続レベル認証が必要な場合は、IBM WebSphere MQ で構成されるクライアント認証出口を作成できます。

IBM WebSphere MQ on z/OS に接続するときにユーザーを認証するには、セキュリティー出口を使用する必要があります。

## **XMSC\_VERSION**

**データ型:**  
     ストリング

**プロパティ:**  
     ConnectionMetaData

XMS クライアントのバージョン ID です。このプロパティは読み取り専用です。

## **XMSC\_WMQ\_BROKER\_CONTROLQ**

**データ型:**  
     ストリング

**プロパティ:**  
     ConnectionFactory

ブローカーが使用する制御キューの名前です。

プロパティのデフォルト値は `SYSTEM.BROKER.CONTROL.QUEUE` です。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## **XMSC\_WMQ\_BROKER\_PUBQ**

**データ型:**  
     ストリング

**プロパティ:**  
     ConnectionFactory

ブローカーによってモニターされているキューの名前で、このキューでは、アプリケーションが発行したメッセージをアプリケーション自体が送信します。

プロパティのデフォルト値は `SYSTEM.BROKER.DEFAULT.STREAM` です。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## **XMSC\_WMQ\_BROKER\_QMGR**

**データ型:**  
     ストリング

**プロパティ:**  
     ConnectionFactory

ブローカーの接続先となるキュー・マネージャーの名前です。

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## **XMSC\_WMQ\_BROKER\_SUBQ**

### **データ型:**

ストリング

### **プロパティ:**

ConnectionFactory

非永続メッセージ・コンシューマー用のサブスクライバー・キューの名前です。

以下の文字で始まる必要があるサブスクライバー・キューの名前です。

SYSTEM.JMS.ND.

すべての非永続メッセージ・コンシューマーがサブスクライバー・キューを共有するには、共有キューの完全な名前を指定します。アプリケーションが非永続メッセージ・コンシューマーを作成できるようにするには、指定した名前のキューが事前に存在する必要があります。

各非永続メッセージ・コンシューマーが自身の排他的サブスクライバー・キューからメッセージを検索するようにしたい場合は、アスタリスク (\*)で終わるキュー名を指定します。その後、アプリケーションが非永続メッセージ・コンシューマーを作成すると、XMS クライアントは、メッセージ・コンシューマーが排他使用するための動的キューを作成します。XMS クライアントは、このプロパティの値を使用して、動的キューを作成するとき使用するオブジェクト記述子の **DynamicQName** フィールドの内容を設定します。

このプロパティのデフォルト値は SYSTEM.JMS.ND.SUBSCRIBER.QUEUE で、これは、XMS は、デフォルトでは共有キューの方法を使用するという意味です。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## **XMSC\_WMQ\_BROKER\_VERSION**

### **データ型:**

System.Int32

### **プロパティ:**

ConnectionFactory および Destination

### **URI で使用される名前:**

brokerVersion

接続または宛先に合わせてアプリケーションが使用するブローカーのタイプです。このプロパティを設定できるのは、内容がトピックである宛先のみです。

プロパティの有効値は以下のとおりです。

### **有効値**

### **意味**

XMSC\_WMQ\_BROKER\_V1

アプリケーションは、IBM WebSphere MQ publish/subscribe ブローカーを使用しています。

IBM WebSphere MQ publish/subscribe から WebSphere Event Broker または WebSphere Message Broker にマイグレーションしたが、アプリケーションを未変更だった場合にも、アプリケーションはこの値を使用できます。

XMSC\_WMQ\_BROKER\_V2

アプリケーションは、WebSphere Event Broker または WebSphere Message Broker のブローカーを使用しています。

XMSC\_WMQ\_BROKER\_UNSPECIFIED

ブローカーをバージョン 6 からバージョン 7 にマイグレーションした後で、RFH2 ヘッダーを使用できなくなるように、このプロパティを設定します。移行した後、このプロパティは関係なくなります。

接続ファクトリーのデフォルト値は `XMSC_WMQ_BROKER_UNSPECIFIED` ですが、デフォルトでは、このプロパティは宛先に設定されていません。このプロパティを宛先に設定すると、接続ファクトリーのプロパティによって指定された値は無効になります。

## **`XMSC_WMQ_CCDTURL`**

**データ型:**  
System.String

**プロパティ:**  
ConnectionFactory

クライアント・チャネル定義テーブルを含むファイルの名前および場所を識別すると同時に、そのファイルへのアクセス方法も指定する、URL (Uniform Resource Locator)。

デフォルトでは、このプロパティは設定されません。

## **`XMSC_WMQ_CCSID`**

**データ型:**  
System.Int32

**プロパティ:**  
Destination

**URI で使用される名前:**  
CCSID

XMS クライアントがメッセージを宛先に転送する場合、メッセージの本文中の一連の文字データに設定されるコード化文字セットの ID (CCSID)、つまりコード・ページです。個々のメッセージに対して設定した場合、このプロパティによって宛先に指定された CCSID は、[JMS IBM CHARACTER SET](#) プロパティによって無効になります。

このプロパティのデフォルト値は 1208 です。

このプロパティが関連するのは宛先に送信されたメッセージのみであり、宛先から受信したメッセージには関連しません。

## **`XMSC_WMQ_CHANNEL`**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

接続に使用するチャンネルの名前です。

デフォルトでは、このプロパティは設定されていません。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが関連します。

## **`XMSC_WMQ_CLIENT_RECONNECT_OPTIONS`**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

このプロパティでは、このファクトリーで作成する新しい接続のクライアント再接続オプションを指定します。これは、XMSC 内にあり、以下のいずれかです。

- `WMQ_CLIENT_RECONNECT_AS_DEF` (デフォルト)。mqclient.ini ファイルで指定されている値を使用します。その値を設定するには、Channels スタンザにある **DefRecon** プロパティを使用します。以下のいずれかに設定できます。

1. はい。WMQ\_CLIENT\_RECONNECT オプションの動作になります。
  2. No. デフォルト。再接続オプションを指定しません。
  3. QMGR。WMQ\_CLIENT\_RECONNECT\_Q\_MGR オプションの動作になります。
  4. DISABLED。WMQ\_CLIENT\_RECONNECT\_DISABLED オプションの動作になります。
- WMQ\_CLIENT\_RECONNECT。接続名リストで指定されているキュー・マネージャーのいずれかに再接続します。
  - WMQ\_CLIENT\_RECONNECT\_Q\_MGR。元の接続先と同じキュー・マネージャーに再接続します。接続しようとする対象のキュー・マネージャー (接続名リストで指定されているキュー・マネージャー) の QMID が元の接続先のキュー・マネージャーの QMID とは異なる場合は、MQRC\_RECONNECT\_QMID\_MISMATCH を返します。
  - WMQ\_CLIENT\_RECONNECT\_DISABLED。再接続が無効になります。

### **XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT**

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

このプロパティでは、クライアント接続が再接続を試みる期間を秒単位で指定します。

クライアントは、この期間、再接続を試みた後に、MQRC\_RECONNECT\_FAILED で失敗します。このプロパティのデフォルト設定は XMSC.WMQ\_CLIENT\_RECONNECT\_TIMEOUT\_DEFAULT です。

このプロパティのデフォルト値は 1800 です。

### **XMSC\_WMQ\_CONNECTION\_MODE**

**データ型:**

System.Int32

**プロパティ:**

ConnectionFactory

アプリケーションがキュー・マネージャーに接続する場合のモードです。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WMQ_CM_BINDINGS	パフォーマンス最適化のための、キュー・マネージャーへのバインディング・モードでの接続。この値は、C/C++ の場合のデフォルト値です。
XMSC_WMQ_CM_CLIENT	完全管理スタックを実現するための、キュー・マネージャーへのクライアント・モードでの接続。この値は、.NET の場合のデフォルト値です。
XMSC_WMQ_CM_CLIENT_UNMANAGED (.NET のみ)	強制的に非管理クライアント・スタックとなるキュー・マネージャーへの接続。

### **関連概念**

ドット・ネットにおける管理操作および非管理操作

管理コードは、ドット・ネット 共通言語ランタイム環境の中で排他的に実行され、そのランタイムによって提供されるサービスに完全に依存します。アプリケーションが非管理に分類されるのは、そのアプリケーションの一部がドット・ネット 共通言語ランタイム環境の外部で実行されるか、または外部にあるサービスを呼び出す場合です。

## **XMSC\_WMQ\_CONNECTION\_NAME\_LIST**

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

このプロパティでは、接続で障害が発生した後にクライアントが再接続を試みる対象のホストを指定します。

接続名リストは、ホスト/IP ポートのペアのコンマ区切りリストです。このプロパティのデフォルト設定は WMQ\_CONNECTION\_NAME\_LIST\_DEFAULT です。

例えば、127.0.0.1(1414),host2.example.com(1400) のようになります。

このプロパティのデフォルト設定は localhost(1414) です。

## **XMSC\_WMQ\_DUR\_SUBQ**

**データ型:**

ストリング

**プロパティ:**

Destination

宛先からメッセージを受信している永続サブスクライバー用のサブスクライバー・キューの名前です。このプロパティを設定できるのは、内容がトピックである宛先のみです。

以下の文字で始まる必要があるサブスクライバー・キューの名前です。

SYSTEM.JMS.D.

すべての永続サブスクライバーがサブスクライバー・キューを共有するには、共有キューの完全な名前を指定します。アプリケーションが永続サブスクライバーを作成できるようにするには、指定した名前のキューが事前に存在する必要があります。

各永続サブスクライバーが自身の排他的サブスクライバー・キューからメッセージを検索するには、末尾にアスタリスク (\*) を付けるキュー名を指定します。その後、アプリケーションが永続サブスクライバーを作成すると、XMS クライアントは、永続サブスクライバーが排他使用するための動的キューを作成します。XMS クライアントは、このプロパティの値を使用して、動的キューを作成するとき使用するオブジェクト記述子の **DynamicQName** フィールドの内容を設定します。

このプロパティのデフォルト値は SYSTEM.JMS.D.SUBSCRIBER.QUEUE で、これは、XMS は、デフォルトでは共有キューの方法を使用するという意味です。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## **XMSC\_WMQ\_ENCODING**

**データ型:**

System.Int32

**プロパティ:**

Destination

XMS クライアントがメッセージを宛先に転送する場合、メッセージの本文中の数値データをどのように表記するかを表します。個々のメッセージに対して設定した場合、このプロパティによって宛先に指定されたエンコード方式は、**JMS IBM\_ENCODING** プロパティによって無効になります。このプロパティは、2 進整数、パック 10 進数、および浮動小数点数の表記を指定します。

このプロパティの有効値は、メッセージ記述子の **Encoding** フィールドで指定できる値と同じです。

アプリケーションは、以下の名前付き定数を使用してプロパティを設定できます。

**名前付き定数**

MQENC\_INTEGER\_NORMAL

**意味**

標準の整数エンコード方式

名前付き定数	意味
MQENC_INTEGER_REVERSED	逆方向の整数エンコード方式
MQENC_DECIMAL_NORMAL	標準のパック 10 進エンコード方式
MQENC_DECIMAL_REVERSED	逆方向のパック 10 進エンコード方式
MQENC_FLOAT_IEEE_NORMAL	標準の IEEE 浮動小数点エンコード方式
MQENC_FLOAT_IEEE_REVERSED	逆方向の IEEE 浮動小数点エンコード方式
MQENC_FLOAT_S390	z/OS アーキテクチャーの浮動小数点エンコード
MQENC_NATIVE	ネイティブのマシン・エンコード方式

プロパティの値を設定するために、アプリケーションは、以下に示す 3 つの定数を加算できます。

- 2 進整数の表記を指定するための、名前が MQENC\_INTEGER で始まる定数
- パック 10 進整数の表記を指定するための、名前が MQENC\_DECIMAL で始まる定数
- 浮動小数点数の表記を指定するための、名前が MQENC\_FLOAT で始まる定数

あるいは、アプリケーションは、その値が環境に依存する MQENC\_NATIVE にプロパティを設定できます。

このプロパティのデフォルト値は MQENC\_NATIVE です。

このプロパティが関連するのは宛先に送信されたメッセージのみであり、宛先から受信したメッセージには関連しません。

## **XMSC\_WMQ\_FAIL\_IF\_QUIESCE**

**データ型:**

System.Int32

**プロパティ:**

ConnectionFactory および Destination

**URI で使用される名前:**

failIfQuiesce

アプリケーションの接続先のキュー・マネージャーが静止状態である場合、特定のメソッドの呼び出しが失敗するかどうかを表します。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WMQ_FIQ_YES	キュー・マネージャーが静止状態である場合、特定のメソッドの呼び出しは失敗します。キュー・マネージャーが静止していることをアプリケーションが検出した場合、アプリケーションはその即時タスクを完了して接続を終了し、キュー・マネージャーを停止できます。
XMSC_WMQ_FIQ_NO	キュー・マネージャーは静止状態であるため、メソッドの呼び出しは失敗しません。この値を指定すると、アプリケーションはキュー・マネージャーが静止していることを検出できません。アプリケーションはキュー・マネージャーに対する操作を続行する可能性があるため、キュー・マネージャーの停止を防止する場合があります。

接続ファクトリーのデフォルト値は XMSC\_WMQ\_FIQ\_YES ですが、デフォルトでは、このプロパティは宛先に設定されていません。このプロパティを宛先に設定すると、接続ファクトリーのプロパティによって指定された値は無効になります。

## **XMSC\_WMQ\_MESSAGE\_BODY**

**データ型:**

System.Int32

#### プロパティ:

Destination

このプロパティは、XMS アプリケーションが IBM WebSphere MQ メッセージの MQRFH2 をメッセージ・ペイロードの一部として (つまり、メッセージ本体の一部として) 処理するかどうかを決定します。

**注:** メッセージを宛先に送信すると、XMSC\_WMQ\_MESSAGE\_BODY プロパティは、既存の XMS Destination プロパティ XMSC\_WMQ\_TARGET\_CLIENT を置き換えます。

このプロパティの有効な値は以下のとおりです。

#### **XMSC\_WMQ\_MESSAGE\_BODY\_JMS**

**Receive:** インバウンド XMS メッセージのタイプと本文は、受信した IBM WebSphere MQ メッセージ内の MQRFH2 (存在する場合) または MQMD (MQRFH2 がいない場合) の内容によって決定されます。

**Send:** アウトバウンド XMS メッセージの本文には、XMS メッセージのプロパティとヘッダー・フィールドに基づいて、前に付加され、自動生成された MQRFH2 ヘッダーが含まれています。

#### **XMSC\_WMQ\_MESSAGE\_BODY\_MQ**

**Receive:** 受信した IBM WebSphere MQ メッセージの内容や受信した MQMD の形式フィールドに関係なく、インバウンド XMS メッセージのタイプは常に `ByteMessage` です。XMS メッセージの本文は、下位層のメッセージング・プロバイダー API 呼び出しによって戻される、未変更のメッセージ・データです。メッセージ本文内のデータの文字セットとエンコードは、MQMD の `CodedCharSetId` フィールドと `Encoding` フィールドによって決定されます。メッセージ本文内のデータの形式は、MQMD の `Format` フィールドによって決定されます。

**Send:** アウトバウンド XMS メッセージの本文には、アプリケーション・ペイロードがそのまま含まれています。自動生成された IBM WebSphere MQ ヘッダーは本文に追加されません。

#### **XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED**

**Receive:** XMS クライアントは、このプロパティに適した値を決定します。受信パスの場合、この値は、`WMQ_MESSAGE_BODY_JMS` プロパティの値になります。

**Send:** XMS クライアントは、このプロパティに適した値を決定します。送信パスの場合、この値は、`XMSC_WMQ_TARGET_CLIENT` プロパティの値になります。

デフォルトでは、このプロパティは `XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED` に設定されます。

#### **XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT**

##### データ型:

`System.Int32`

##### プロパティ:

Destination

XMS アプリケーションによって設定されるメッセージ・コンテキストのレベルを決定します。このプロパティが有効となるためには、アプリケーションが適切なコンテキスト権限を持って実行されていなければなりません。

このプロパティの有効な値は以下のとおりです。

#### **XMSC\_WMQ\_MDCTX\_DEFAULT**

アウトバウンド・メッセージでは、`MQOPEN` API 呼び出しと `MQPMO` 構造は、明示的なメッセージ・コンテキスト・オプションを指定しません。

#### **XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT**

`MQOPEN` API 呼び出しは、メッセージ・コンテキスト・オプション `MQOO_SET_IDENTITY_CONTEXT` を指定し、`MQPMO` 構造は `MQPMO_SET_IDENTITY_CONTEXT` を指定します。

#### **XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT**

`MQOPEN` API 呼び出しはメッセージ・コンテキスト・オプション `MQOO_SET_ALL_CONTEXT` を指定し、`MQPMO` 構造は `MQPMO_SET_ALL_CONTEXT` を指定します。

デフォルトでは、このプロパティは `XMSC_WMQ_MDCTX_DEFAULT` に設定されます。

注: アプリケーションがシステム統合バスに接続している場合、このプロパティーは関係ありません。

望ましい効果を得るには、以下のプロパティーでは、メッセージの送信時に、  
XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT プロパティーが  
XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT プロパティー値または  
XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT プロパティー値に設定されている必要があります。

- JMS\_IBM\_MQMD\_USERIDENTIFIER
- JMS\_IBM\_MQMD\_ACCOUNTINGTOKEN
- JMS\_IBM\_MQMD\_APPLIDENTITYDATA

望ましい効果を得るには、以下のプロパティーでは、メッセージの送信時に、  
XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT プロパティーが XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT プロ  
パティー値に設定されている必要があります。

- 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**

データ型:  
System.Int32

プロパティー:  
Destination

このプロパティーは、XMS アプリケーションが MQMD フィールドの値を抽出できるかどうかを決定しま  
す。

このプロパティーの有効な値は以下のとおりです。

#### **XMSC\_WMQ\_READ\_ENABLED\_NO**

メッセージの送信時に、送信されたメッセージの JMS\_IBM\_MQMD\* プロパティーは、MQMD での更新  
済みのフィールド値を反映するように更新されません。

送信側が JMS\_IBM\_MQMD\* プロパティーの一部またはすべてを設定しておいた場合でも、メッセージ  
の受信時には、受信したメッセージでこれらのプロパティーはいずれも有効になりません。

#### **XMSC\_WMQ\_READ\_ENABLED\_YES**

メッセージの送信時に、送信したメッセージの JMS\_IBM\_MQMD\* プロパティーはすべて、MQMD の更  
新後のフィールド値に合わせて更新されます (送信側が明示的に設定しなかったプロパティーも含めて  
そのようになります)。

メッセージの受信時には、受信したメッセージですべての JMS\_IBM\_MQMD\* プロパティーが有効にな  
ります (送信側が明示的に設定しなかったプロパティーも含めてそのようになります)。

デフォルトでは、このプロパティーは XMSC\_WMQ\_READ\_ENABLED\_NO に設定されます。

### **XMSC\_WMQ\_MQMD\_WRITE\_ENABLED**

データ型:  
System.Int32

プロパティー:  
Destination

このプロパティーは、XMS アプリケーションが MQMD フィールドの値を書き込むことができるかどうかを  
決定します。

このプロパティーの有効な値は以下のとおりです。

### **XMSC\_WMQ\_WRITE\_ENABLED\_NO**

JMS\_IBM\_MQMD\* プロパティはすべて無視され、その値は基礎となる MQMD 構造にコピーされません。

### **XMSC\_WMQ\_WRITE\_ENABLED\_YES**

JMS\_IBM\_MQMD\* プロパティは処理されます。それらの値は基礎となる MQMD 構造にコピーされます。

デフォルトでは、このプロパティは XMSC\_WMQ\_WRITE\_ENABLED\_NO に設定されます。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED**

データ型:

System.Int32

プロパティ:

Destination

このプロパティは、メッセージ・プロデューサーが、非同期書き込みを使用して、この宛先にメッセージを送信できるかどうかを決定します。

このプロパティの有効な値は以下のとおりです。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST**

キュー定義またはトピック定義を参照することによって、非同期書き込みが許可されるかどうかを決定します。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_Q\_DEF**

キュー定義を参照することによって非同期書き込みが許可されるかどうかを判別します。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_TOPIC\_DEF**

トピック定義を参照することによって非同期書き込みが許可されるかどうかを判別します。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_DISABLED**

非同期書き込みは許可されない。

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_ENABLED**

非同期書き込みは許可される。

デフォルトでは、このプロパティは XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST に設定されます。

注: アプリケーションがシステム統合バスに接続している場合、このプロパティは関係ありません。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED**

データ型:

System.Int32

プロパティ:

Destination

このプロパティは、メッセージ・コンシューマーとキュー・ブラウザーが、先行読み取りを使用して、非永続の非トランザクション・メッセージを受信する前に、この宛先から内部バッファーにこれらのメッセージを取得できるかどうかを決定します。

このプロパティの有効な値は以下のとおりです。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF**

キュー定義を参照することによって、先行読み取りが許可されるかどうかを決定します。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TOPIC\_DEF**

トピック定義を参照することによって、先行読み取りが許可されるかどうかを決定します。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST**

キュー定義またはトピック定義を参照することによって、先行読み取りが許可されるかどうかを決定します。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED**

メッセージのコンシュームまたは参照時には先行読み取りは許可されません。

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_ENABLED**

先読みは許可される。

デフォルトでは、このプロパティは XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST に設定されます。

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY**

データ型:

System.Int32

プロパティ:

Destination

このプロパティは、非同期メッセージ・リスナーに配信されているメッセージについて、メッセージ・コンシューマーがクローズされたときに、内部先行読み取りバッファ内のメッセージがどうなるかを決定します。

このプロパティは、宛先からメッセージをコンシュームするときにキューのクローズ・オプションを指定する際に適用され、宛先にメッセージを送信する際は適用されません。

参照中にメッセージはまだキューで使用できるため、このプロパティはキュー・ブラウザーでは無視されます。

このプロパティの有効な値は以下のとおりです。

#### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT**

現行のメッセージ・リスナーの呼び出しのみが完了して戻ります。この場合、内部先行読み取りバッファにメッセージが残る可能性があり、それらは廃棄されます。

#### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_ALL**

内部先行読み取りバッファ内のメッセージがすべてアプリケーションのメッセージ・リスナーに配信されてから戻ります。

デフォルトでは、このプロパティは XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT に設定されます。

注:

#### • アプリケーションの異常終了

XMS アプリケーションが不意に終了すると、先行読み取りバッファ内のメッセージはすべて失われます。

#### • トランザクションへの影響

アプリケーションでトランザクションが使用される場合は、先行読み取りは使用不可にされます。そのため、アプリケーションでは、トランザクション化セッションを使用する場合は、動作の相違点は認識されません。

#### • セッション肯定応答モードの影響

肯定応答モードが XMSC\_AUTO\_ACKNOWLEDGE または XMSC\_DUPS\_OK\_ACKNOWLEDGE になっていると、非トランザクション化セッションで先行読み取りが有効になります。トランザクション化セッションか未トランザクション化セッションかに関係なく、セッション肯定応答モードが XMSC\_CLIENT\_ACKNOWLEDGE の場合は、先行読み取りは使用不可になります。

#### • キュー・ブラウザーとキュー・ブラウザー・セレクターへの影響

XMS アプリケーションで使用されるキュー・ブラウザーとキュー・ブラウザー・セレクターでは、先行読み取りによりパフォーマンスが向上します。キュー・ブラウザーを閉じて、パフォーマンスが低下

することはありません。その後の操作でも、引き続きキューのメッセージを使用できるからです。先行読み取りによってパフォーマンスが向上するという以外に、キュー・ブラウザーとキュー・ブラウザー・セレクターに対する影響はありません。

- **WebSphere メッセージ・ブローカー V6 以前のキュー・マネージャーへの先行読み取り宛先プロパティの影響**

XMS アプリケーションで WebSphere メッセージ・ブローカー V6 キュー・マネージャーが使用される場合に、宛先プロパティ `XMSC_WMQ_READ_AHEAD_ALLOWED` と `XMSC_WMQ_READ_AHEAD_CLOSE_POLICY` を指定すると、指定した値を使用できなくなります。これらの宛先プロパティ値は、通知されることなく無視され、アプリケーションは先行読み取りを行わずに作業を続行します。V6 キュー・マネージャーでの使用時にはエラーはスローされません。

## **`XMSC_WMQ_HOST_NAME`**

### **データ型:**

ストリング

### **プロパティ:**

ConnectionFactory

キュー・マネージャーを実行するシステムのホスト名または IP アドレスです。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが使用されます。このプロパティは、キュー・マネージャーを識別するために、`XMSC_WMQ_PORT` プロパティと一緒に使用します。

このプロパティのデフォルト値は `localhost` です。

## **`XMSC_WMQ_LOCAL_ADDRESS`**

### **データ型:**

ストリング

### **プロパティ:**

ConnectionFactory

キュー・マネージャーへの接続の場合、このプロパティは、使用するローカル・ネットワーク・インターフェース、または使用するローカル・ポート (1 つまたは一定範囲)、あるいはその両方を指定します。

このプロパティの値は、次の形式のストリングです。

`[host_name][([low_port],[high_port])]`

変数の意味は以下のとおりです。

### **`host_name`**

接続に使用するローカル・ネットワーク・インターフェースのホスト名または IP アドレスです。

この情報の入力が必要なのは、アプリケーションを実行しているシステムに複数のネットワーク・インターフェースがあり、接続にどのインターフェースを使用する必要があるかを指定できることが必要な場合に限ります。システムが備えるネットワーク・インターフェースが 1 つのみである場合、使用できるのはそのインターフェースのみです。システムに複数のネットワーク・インターフェースがあり、どのインターフェースを使用するかを指定していない場合、インターフェースはランダムに選択されます。

### **`low_port`**

接続に使用するローカル・ポートの数です。

`high_port` も指定した場合、`low_port` は一連のポート番号のうち最小のポート番号と解釈されます。

### **`high_port`**

一連のポート番号のうち最大のポート番号を表します。指定した範囲内のいずれかのポートを接続に使用する必要があります。

このストリングの最大長は 48 文字です。

プロパティの有効値の例の一部を以下に示します。

JUPITER  
9.20.4.98  
JUPITER(1000)  
9.20.4.98(1000,2000)  
(1000)  
(1000,2000)

デフォルトでは、このプロパティは設定されていません。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが関連します。

## ***XMSC\_WMQ\_MESSAGE\_SELECTION***

データ型:

System.Int32

プロパティ:

ConnectionFactory

メッセージの選択を XMS クライアントで行うか、ブローカーで行うかを判別します。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WMQ_MSEL_CLIENT	メッセージの選択は XMS クライアントによって行われます。
XMSC_WMQ_MSEL_BROKER	ブローカーがメッセージ選択を行う。

デフォルト値は XMSC\_WMQ\_MSEL\_CLIENT です。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

XMSC\_WMQ\_BROKER\_VERSION プロパティを XMSC\_BROKER\_V1 に設定している場合、ブローカーによるメッセージの選択はサポートされません。

## ***XMSC\_WMQ\_MSG\_BATCH\_SIZE***

データ型:

System.Int32

プロパティ:

ConnectionFactory

非同期のメッセージ配信機能を使用する場合に 1 バッチ内のキューから取り出すメッセージの最大数を表します。

アプリケーションが特定の条件下で非同期のメッセージ配信機能を使用している場合、XMS クライアントは、1 バッチのメッセージをキューから取り出してから、各メッセージをアプリケーションに個別に転送します。このプロパティでは、バッチ内に収容できるメッセージの最大数が指定されます。

このプロパティの値は正の整数で、デフォルト値は 10 です。このプロパティを他の値に設定することを検討するのは、対応しなければならないパフォーマンスに関する具体的な問題がある場合に限るようにしてください。

アプリケーションがネットワークを介してキュー・マネージャーに接続されている場合、このプロパティの値を大きくすると、ネットワークのオーバーヘッドを削減して応答時間を短縮できますが、クライアントのシステムでメッセージを保管するのに必要なメモリーの量は増加します。逆に、このプロパティの値を小さくすると、ネットワークのオーバーヘッドは増大し、応答時間は長くなりますが、メッセージを保管するのに必要なメモリーの量を削減できます。

## ***XMSC\_WMQ\_POLLING\_INTERVAL***

データ型:

System.Int32

#### プロパティ:

ConnectionFactory

セッション内の各メッセージ・リスナーのキューに適切なメッセージがない場合、この値は、各メッセージ・リスナーがそのキューから再度メッセージを読み取ろうとするまでに経過する最大の時間間隔(ミリ秒)になります。

セッション内のいずれのメッセージ・リスナーでも適切なメッセージがない状態が頻繁に発生する場合は、このプロパティの値を大きくすることを考えてください。

このプロパティの値は正の整数です。デフォルト値は 5000 です。

### **XMSC\_WMQ\_PORT**

#### データ型:

System.Int32

#### プロパティ:

ConnectionFactory

キュー・マネージャーが着信要求を listen するポートの数です。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが使用されます。このプロパティは、キュー・マネージャーを識別するために、[XMSC\\_WMQ\\_HOST\\_NAME](#) プロパティと一緒に使用します。

プロパティのデフォルト値は `XMSC_WMQ_DEFAULT_CLIENT_PORT`、または 1414 です。

### **XMSC\_WMQ\_PROVIDER\_VERSION**

#### データ型:

ストリング

#### プロパティ:

ConnectionFactory

アプリケーションの接続先のキュー・マネージャーのバージョン、リリース、モディフィケーション・レベル、およびフィックスパック。このプロパティの有効な値は以下のとおりです。

- 指定なし

または、以下のいずれかの形式のストリング

- V.R.M.F
- V.R.M
- V.R
- V

ここで、V、R、M、および F は、0 以上の整数値です。

7 以上の値を指定する場合は、IBM WebSphere MQ バージョン 7.0 のキュー・マネージャーに接続するために IBM WebSphere MQ バージョン 7.0 の ConnectionFactory としてこのバージョンを使用するという意味になります。7 より低い値(例えば 6.0.2.0)は、バージョン 7.0 より前のキュー・マネージャーと共に使用することを示します。デフォルト値である「UNSPECIFIED」を使用すると、キュー・マネージャーの機能に基づいて適用可能なプロパティと使用可能な機能を判別して、任意のレベルのキュー・マネージャーに接続できます。

デフォルトでは、このプロパティは「UNSPECIFIED」に設定されています。

#### 注:

- XMSC\_WMQ\_PROVIDER\_VERSION が 6 に設定されている場合、ソケット共有は行われません。2.
- XMSC\_WMQ\_PROVIDER\_VERSION が 7 に設定されていて、チャンネルのサーバー SHARECNV が 0 に設定されていると、接続は失敗します。

- XMSC\_WMQ\_PROVIDER\_VERSION が UNSPECIFIED に設定されていて、SHARECNV が 0 に設定されている場合は、IBM WebSphere MQ バージョン 7.0 に固有の機能は使用不可になります。

IBM WebSphere MQ クライアントのバージョンは、XMS クライアント・アプリケーションで IBM WebSphere MQ バージョン 7.0 の固有の機能を使用できるかどうかに関して重要な役割を果たします。以下の表に、動作の説明を示します。

注：システム・プロパティ XMSC\_WMQ\_OVERRIDEPROVIDERVERSION は、XMSC\_WMQ\_PROVIDER\_VERSION プロパティより優先されます。このプロパティは、接続ファクトリー設定を変更できない場合に使用できます。

#	XMSC_WMQ_PROVIDER_VERSION	IBM WebSphere MQ クライアントのバージョン	IBM WebSphere MQ バージョン 7.0 の機能
1	指定なし	7	ON
2	指定なし	6	OFF
3	7	7	ON
4	7	6	例外
5	6	6	OFF
6	6	7	OFF

### **XMSC\_WMQ\_PUB\_ACK\_INTERVAL**

データ型:

System.Int32

プロパティ:

ConnectionFactory

XMS クライアントがブローカーから肯定応答を要求するまでにパブリッシャーによって発行されるメッセージの数です。

このプロパティの値を小さくすると、クライアントによる肯定応答の要求頻度が高くなるため、パブリッシャーのパフォーマンスは低下します。この値を大きくすると、ブローカーに障害が発生した場合にクライアントが例外をスローするのに要する時間が長くなります。

このプロパティの値は正の整数です。デフォルト値は 25 です。

### **XMSC\_WMQ\_QMGR\_CCSID**

データ型:

System.Int32

プロパティ:

ConnectionFactory

メッセージ・キュー・インターフェース (MQI) で定義された文字データのフィールドが XMS クライアントと IBM WebSphere MQ クライアントの間で交換されるコード化文字セットまたはコード・ページの ID (CCSID)。このプロパティは、メッセージ本文の文字データのストリングには適用されません。

XMS アプリケーションがキュー・マネージャーにクライアント・モードで接続すると、XMS クライアントは IBM WebSphere MQ クライアントにリンクします。この 2 つのクライアント間で交換される情報には、MQI で定義された文字データのフィールドが含まれます。通常的环境下では、IBM WebSphere MQ クライアントは、そのクライアントが動作しているシステムのコード・ページでこれらのフィールドが記述されていることを前提としています。XMS クライアントが異なるコード・ページでこれらのフィールドを提供する場合や、これらのフィールドを異なるコード・ページで受信することが予想される場合は、このプロパティを設定して IBM WebSphere MQ クライアントに通知する必要があります。

IBM WebSphere MQ クライアントがこれらの文字データ・フィールドをキュー・マネージャーに転送した場合は、必要に応じて、これらのフィールド内のデータを、キュー・マネージャーが使用するコード・ペ

ージに変換する必要があります。同様に、IBM WebSphere MQ クライアントがこれらのフィールドをキュー・マネージャーから受信した場合は、必要に応じて、これらのフィールド内のデータを、XMS クライアントがデータを受信する場合に想定しているコード・ページに変換する必要があります。IBM WebSphere MQ クライアントは、このプロパティを使用してこれらのデータ変換を実行します。

デフォルトでは、このプロパティは設定されていません。

このプロパティを設定することは、ネイティブの IBM WebSphere MQ クライアント・アプリケーションをサポートしている IBM WebSphere MQ クライアントに対して MQCCSID 環境変数を設定することと同等です。この環境変数の詳細については、「*IBM WebSphere MQ* クライアント」を参照してください。

## ***XMSC\_WMQ\_QUEUE\_MANAGER***

**データ型:**  
     文字列

**プロパティ:**  
     ConnectionFactory

接続先となるキュー・マネージャーの名前です。

デフォルトでは、このプロパティは設定されていません。

## ***XMSC\_WMQ\_RECEIVE\_CCSID***

キュー・マネージャー・メッセージ変換のターゲット CCSID を設定する宛先プロパティ。  
XMSC\_WMQ\_RECEIVE\_CONVERSION が WMQ\_RECEIVE\_CONVERSION\_QMGR に設定されていなければ、この値は無視されます。

**データ型:**  
     整数

**値:**  
     任意の正整数。

デフォルト値は 1208 です。

## ***XMSC\_WMQ\_RECEIVE\_CONVERSION***

キュー・マネージャーによりデータ変換を実行するかどうかを決定する宛先プロパティ。

**データ型:**  
     整数

**値:**  
XMSC\_WMQ\_RECEIVE\_CONVERSION\_CLIENT\_MSG (デフォルト): XMS クライアントでのみデータ変換を実行します。変換では常にコード・ページ 1208 を使用します。  
XMSC\_WMQ\_RECEIVE\_CONVERSION\_QMGR: XMS クライアントにメッセージを送信する前にキュー・マネージャーでデータ変換を実行します。

## ***XMSC\_WMQ\_RECEIVE\_EXIT***

**データ型:**  
     文字列

**プロパティ:**  
     ConnectionFactory

実行するチャンネル受信出口を示します。

このプロパティの値は、チャンネル受信出口を示す文字列で、その形式は次のとおりです。

**libraryName(entryPointName)**

ここで、

- **libraryName** は、管理出口 .dll の絶対パスです。

- `entryPointName` は、名前空間で修飾されるクラス名です。

`C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)` に例を示します

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続している場合のみです。また、管理出口のみがサポートされます。

## ***XMSC\_WMQ\_RECEIVE\_EXIT\_INIT***

**データ型:**

    文字列

**プロパティ:**

    ConnectionFactory

チャンネル受信出口が呼び出されたときにこの出口に渡されるユーザー・データです。

プロパティの値は文字列です。デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続し、[233 ページの『XMSC\\_WMQ\\_RECEIVE\\_EXIT』](#) プロパティが設定されている場合のみです。

## ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER***

**データ型:**

    文字列

**プロパティ:**

    ConnectionFactory

このプロパティは、接続先のキュー・マネージャーの名前を取得するために使用します。

CCDT (クライアント・チャンネル定義テーブル) と一緒に使用する場合、この名前は、接続ファクトリーで指定されているキュー・マネージャー名とは異なる可能性があります。

## ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER\_ID***

**データ型:**

    文字列

**プロパティ:**

    ConnectionFactory

接続後に、このプロパティにはキュー・マネージャーの ID が設定されます。

## ***XMSC\_WMQ\_SECURITY\_EXIT***

**データ型:**

    文字列

**プロパティ:**

    ConnectionFactory

チャンネル・セキュリティー出口を示します。

このプロパティの値はチャンネル・セキュリティー出口を示す文字列で、その形式は次のとおりです。

**libraryName**(entryPointName)

ここで、

- **libraryName** は、管理出口 .dll の絶対パスです。
- `entryPointName` は、名前空間で修飾されるクラス名です。

例えば、`C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)` となります。

ストリングの最大長は 128 文字です。

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続している場合のみです。また、管理出口のみがサポートされます。

### ***XMSC\_WMQ\_SECURITY\_EXIT\_INIT***

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

チャンネル・セキュリティ出口を呼び出した場合にこの出口に渡されるユーザー・データです。

ユーザー・データのストリングの最大長は 32 文字です。

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続し、[234 ページの『XMSC\\_WMQ\\_SECURITY\\_EXIT』](#) プロパティが設定されている場合のみです。

### ***XMSC\_WMQ\_SEND\_EXIT***

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

チャンネル送信出口を示します。

プロパティの値はストリングです。チャンネル送信出口の形式は次のとおりです。

**libraryName(entryPointName)**

ここで、

- **libraryName** は、管理出口 .dll の絶対パスです。
- **entryPointName** は、名前空間で修飾されるクラス名です。

例えば、 `C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)`

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続している場合のみです。また、管理出口のみがサポートされます。

### ***XMSC\_WMQ\_SEND\_EXIT\_INIT***

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

複数のチャンネル送信出口を呼び出した場合にこれらの出口に渡されるユーザー・データです。

このプロパティの値は、コマンドで区切られた 1 つ以上のユーザー・データ項目から成るストリングです。デフォルトでは、このプロパティは設定されていません。

チャンネル送信出口のシーケンスに渡されるユーザー・データを指定するための規則は、チャンネル受信出口のシーケンスに渡されるユーザー・データを指定するための規則と同じです。したがって、この規則については、[234 ページの『XMSC\\_WMQ\\_RECEIVE\\_EXIT\\_INIT』](#)を参照してください。

このプロパティが関連するのは、アプリケーションが管理クライアント・モードでキュー・マネージャーに接続し、[235 ページの『XMSC\\_WMQ\\_SEND\\_EXIT』](#)プロパティが設定されている場合のみです。

## **XMSC\_WMQ\_SEND\_CHECK\_COUNT**

**データ型:**  
System.Int32

**プロパティ:**  
ConnectionFactory

単一の未処理の XMS セッション内での、非同期書き込みエラーの検査の間に許可する Send 呼び出しの数。デフォルトでは、このプロパティは 0 に設定されます。

## **XMSC\_WMQ\_SHARE\_CONV\_ALLOWED**

**データ型:**  
System.Int32

**プロパティ:**  
ConnectionFactory

チャンネル定義が一致する場合に、クライアント接続が、同じプロセスから同じキュー・マネージャーへの他の最上位 XMS 接続とソケットを共有できるかどうか。このプロパティは、アプリケーション開発、保守、または操作に必要な場合に、別のソケットの接続を完全に分離できるようにするために提供されています。このプロパティを設定すると、基礎となるソケットを共有するように XMS に指示のみをします。単一のソケットを共有する接続の数は示されません。ソケットを共有している接続の数は、IBM WebSphere MQ クライアントと IBM WebSphere MQ サーバー間でネゴシエーションされる SHARECNV 値によって決まります。

アプリケーションは、以下の名前付き定数を設定して、プロパティを設定できます。

- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_FALSE - 複数の接続間でソケットは共有されません。
- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_TRUE - 複数の接続間でソケットは共有されます。

デフォルトでは、このプロパティは XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_ENABLED に設定されます。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが関連します。

## **XMSC\_WMQ\_SSL\_CERT\_STORES**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

キュー・マネージャーとの SSL 接続で使用される証明書取り消しリスト (CRL) を保持するサーバーの位置。このプロパティの値は、1 つ以上の URL をコンマで区切ったリストです。各 URL の形式は次のとおりです。

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

この形式は、基本的な MQJMS 形式と互換性がありますが、一部拡張されています。

serveraddress の部分を空にしても有効です。その場合、XMS は、その値が「localhost」というストリングであるとみなします。

リストの例を以下に示します。

```
myuser/mypassword@ldap://server1.mycom.com:389  
ldap://server1.mycom.com  
ldap://  
ldap://:389
```

ドット・ネットの場合のみ: IBM WebSphere MQ (WMQ\_CM\_CLIENT) への管理対象接続は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる場合があります。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WMQ\_SSL\_CIPHER\_SPEC**

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

**V7.5.0.2** キュー・マネージャーとのセキュア接続で使用する CipherSpec の名前。

次の表に、IBM WebSphere MQ SSL および TLS サポートとともに使用できる Cipher 仕様をリストします。個人用証明書を要求するときに、公開鍵と秘密鍵のペアの鍵サイズを指定します。SSL ハンドシェイク時に使用される鍵のサイズは、表の注記のとおり、CipherSpec によって決定されている場合を除き、証明書に保管されているサイズです。デフォルトでは、このプロパティは設定されません。

CipherSpec 名	使用されるプロトコル	ハッシュ・アルゴリズム	暗号化アルゴリズム	暗号化ビット数	FIPS <sup>1</sup>	Suite B 128 ビット	Suite B 192 ビット
NULL_MD5	SSL 3.0 8	MD5	なし	0	いいえ	いいえ	いいえ
NULL_SHA	SSL 3.0 8	SHA-1	なし	0	いいえ	いいえ	いいえ
RC4_MD5_EXPORT <sup>2</sup>	SSL 3.0 8	MD5	RC4	40	いいえ	いいえ	いいえ
RC4_MD5_US	SSL 3.0 8	MD5	RC4	128	いいえ	いいえ	いいえ
RC4_SHA_US	SSL 3.0 8	SHA-1	RC4	128	いいえ	いいえ	いいえ
RC2_MD5_EXPORT <sup>2</sup>	SSL 3.0 8	MD5	RC2	40	いいえ	いいえ	いいえ
DES_SHA_EXPORT <sup>2</sup>	SSL 3.0 8	SHA-1	DES	56	いいえ	いいえ	いいえ
RC4_56_SHA_EXPORT1024 <sup>3</sup>	SSL 3.0 8	SHA-1	RC4	56	いいえ	いいえ	いいえ
DES_SHA_EXPORT1024 <sup>3</sup>	SSL 3.0 8	SHA-1	DES	56	いいえ	いいえ	いいえ
TRIPLE_DES_SHA_US	SSL 3.0 8	SHA-1	3DES	168	いいえ	いいえ	いいえ
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	AES	128	Yes	いいえ	いいえ
TLS_RSA_WITH_AES_256_CBC_SHA <sup>4</sup>	TLS 1.0	SHA-1	AES	256	Yes	いいえ	いいえ
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	DES	56	いいえ <sup>5</sup>	いいえ	いいえ
TLS_RSA_WITH_3DES_EDE_CBC_SHA <sup>8</sup>	TLS 1.0	SHA-1	3DES	168	Yes	いいえ	いいえ

CipherSpec 名	使用されるプロトコル	ハッシュ・アルゴリズム	暗号化アルゴリズム	暗号化ビット数	FIPS <sup>1</sup>	Suite B 128 ビット	Suite B 192 ビット
FIPS_WITH_DES_CBC_SHA	SSL 3.0	SHA-1	DES	56	いいえ 6	いいえ	いいえ
FIPS_WITH_3DES_EDE_CBC_SHA	SSL 3.0	SHA-1	3DES	168	いいえ 7	いいえ	いいえ
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Yes	いいえ	いいえ
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Yes	いいえ	いいえ
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Yes	いいえ	いいえ
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	AES	256	Yes	いいえ	いいえ
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	いいえ	いいえ	いいえ
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Yes	いいえ	いいえ
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	いいえ	いいえ	いいえ
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Yes	いいえ	いいえ
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Yes	いいえ	いいえ
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Yes	いいえ	いいえ
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Yes	いいえ	いいえ
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Yes	いいえ	いいえ
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Yes	Yes	いいえ
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Yes	いいえ	Yes
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Yes	いいえ	いいえ
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Yes	いいえ	いいえ
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	なし	0	いいえ	いいえ	いいえ
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	なし	0	いいえ	いいえ	いいえ

CipherSpec 名	使用されるプロトコル	ハッシュ・アルゴリズム	暗号化アルゴリズム	暗号化ビット数	FIPS <sup>1</sup>	Suite B 128 ビット	Suite B 192 ビット
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	なし	0	いいえ	いいえ	いいえ
TLS_RSA_WITH_NULL_NULL	TLS 1.2	なし	なし	0	いいえ	いいえ	いいえ
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	いいえ	いいえ	いいえ

**注:**

1. CipherSpec が連邦情報処理標準 (FIPS) 140-2 に準拠しているかどうかを示しています。FIPS の説明、および FIPS 140-2 準拠の操作に WebSphere MQ を構成する方法については、オンラインの IBM WebSphere MQ 製品資料で「連邦情報処理標準 (FIPS)」を参照してください。
2. ハンドシェイクの最大鍵サイズは 512 ビットです。SSL ハンドシェイク時に交換されるどちらかの証明書の鍵サイズが 512 ビットより大きい場合は、ハンドシェイク時に使用するための 512 ビットの一時鍵が生成されます。
3. ハンドシェイクの鍵サイズは 1024 ビットです。
4. WebSphere MQ エクスプローラーが使用する JRE に対して適切な無制限のポリシー・ファイルが適用されていない場合には、この CipherSpec を使用して、WebSphere MQ エクスプローラーからキュー・マネージャーへの安全な接続を確立することはできません。
5. この CipherSpec は、2007 年 5 月 19 日より前は FIPS 140-2 で認証されていました。
6. この CipherSpec は、2007 年 5 月 19 日より前は FIPS 140-2 で認証されていました。FIPS\_WITH\_DES\_CBC\_SHA という名前は歴史的なものであり、この CipherSpec がかつて FIPS 準拠であったという事実を反映しています (ただし、現在は準拠していません)。この CipherSpec は推奨されません。
7. FIPS\_WITH\_3DES\_EDE\_CBC\_SHA という名前は歴史的なものであり、この CipherSpec がかつて FIPS 準拠であったという事実を反映しています (ただし、現在は準拠していません)。この CipherSpec の使用は推奨されません。
8. WebSphere MQ が FIPS 140-2 に準拠した運用のために構成されている場合、この CipherSpec を使用して最大 32 GB までデータを転送できますが、それを超えるとエラー AMQ9288 を出して接続が終了します。このエラーを回避するために、Triple-DES を使用しないか (これは非推奨です)、または FIPS 140-2 構成でこの CipherSpec を使用する際に秘密鍵リセットを有効にします。

**関連概念**

機密保護

[メッセージのデータ保全性](#)

**関連タスク**

[CipherSpec の指定](#)

***XMSC\_WMQ\_SSL\_CIPHER\_SUITE***

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

キュー・マネージャーへの SSL 接続または TLS 接続で使用される CipherSuite の名前。セキュア接続のネゴシエーションで使用されるプロトコルは、指定されている CipherSuite によって異なります。

このプロパティの標準値は以下のとおりです。

- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_DES\_CBC\_SHA

- SSL\_RSA\_FIPS\_WITH\_3DES\_EDE\_CBC\_SHA
- SSL\_RSA\_FIPS\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_WITH\_NULL\_MD5
- SSL\_RSA\_WITH\_NULL\_SHA
- SSL\_RSA\_EXPORT\_WITH\_RC2\_CBC\_40\_MD5
- SSL\_RSA\_EXPORT1024\_WITH\_RC4\_56\_SHA
- SSL\_RSA\_EXPORT\_WITH\_RC4\_40\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_SHA
- SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_128\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_256\_CBC\_SHA
- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

この値は、XMSC\_WMQ\_SSL\_CIPHER\_SPEC に代わるものとして指定できます。

XMSC\_WMQ\_SSL\_CIPHER\_SPEC に空でない値が指定されている場合、XMSC\_WMQ\_SSL\_CIPHER\_SUITE の設定値はこの値によってオーバーライドされます。XMSC\_WMQ\_SSL\_CIPHER\_SPEC に値が指定されていない場合、GSKit に提供される暗号スイートとして XMSC\_WMQ\_SSL\_CIPHER\_SUITE の値が使用されます。その場合、68 ページの『IBM WebSphere MQ キュー・マネージャー への接続のための CipherSuite および CipherSpec 名前マッピング』の記述に従って、CipherSpec で相当する値にマップされます。

XMSC\_WMQ\_SSL\_CIPHER\_SPEC と XMSC\_WMQ\_SSL\_CIPHER\_SUITE のどちらも空の場合、フィールド `pChDef->SSLCipherSpec` にはスペースが入ります。

ドット・ネットの場合のみ: IBM WebSphere MQ (WMQ\_CM\_CLIENT) への管理対象接続は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる場合があります。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WMQ\_SSL\_CRYPTO\_HW**

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

クライアント・システムに接続されている暗号ハードウェアに関する構成詳細情報。

このプロパティの標準値は以下のとおりです。

- GSK\_ACCELERATOR\_RAINBOW\_CS\_OFF
- GSK\_ACCELERATOR\_RAINBOW\_CS\_ON
- GSK\_ACCELERATOR\_NCIPHER\_NF\_OFF
- GSK\_ACCELERATOR\_NCIPHER\_NF\_ON

PKCS11 暗号ハードウェアについては特殊な形式があります (DriverPath、TokenLabel、TokenPassword はユーザー指定ストリング)。

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS は、ストリングの内容を解釈したり変更したりしません。指定された値のうち最大 256 個の 1 バイト文字に相当する部分を MQSCO.CryptoHardware フィールドにコピーします。

ドット・ネットの場合のみ: IBM WebSphere MQ への管理対象接続 (WMQ\_CM\_CLIENT) は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる可能性があります。

デフォルトでは、このプロパティは設定されていません。

## ***XMSC\_WMQ\_SSL\_FIPS\_REQUIRED***

**データ型:**

ブール値

**プロパティ:**

ConnectionFactory

このプロパティの値は、アプリケーションが非 FIPS 準拠の暗号スイートを使用できるかどうかを決定します。このプロパティが true (真) に設定されている場合、クライアント/サーバー接続には FIPS アルゴリズムだけが使用されます。

このプロパティの値は次のとおりです。それらは、MQSCO.FipsRequired の 2 個の標準値に変換されます。

値	説明	MQSCO.FipsRequired で対応する値
false	任意の CipherSpec を使用できます。	MQSSL_FIPS_NO (デフォルト)
true	このクライアント接続に適用する CipherSpec では、FIPS 認証暗号アルゴリズムのみ使用できます。	MQSSL_FIPS_YES

XMS は、MQCONN を呼び出す前に、関係する値を MQSCO.FipsRequired にコピーします。

パラメーター MQSCO.FipsRequired は、IBM WebSphere MQ バージョン 6 からのみ使用可能です。IBM WebSphere MQ バージョン 5.3 の場合、このプロパティが設定されていると、XMS はキュー・マネージャーへの接続を試行せず、代わりに適切な例外をスローします。

ドット・ネットの場合のみ: IBM WebSphere MQ への管理対象接続 (WMQ\_CM\_CLIENT) は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる可能性があります。

## ***XMSC\_WMQ\_SSL\_KEY\_REPOSITORY***

**データ型:**

ストリング

**プロパティ:**

ConnectionFactory

鍵および証明書が保存されている鍵データベース・ファイルの位置。

XMS は、ストリングのうち 256 個までの 1 バイト文字に相当する部分を MQSCO.KeyRepository フィールドにコピーします。IBM WebSphere MQ はこのストリングを絶対パスを含むファイル名と解釈します。

ドット・ネットの場合のみ: IBM WebSphere MQ (WMQ\_CM\_CLIENT) への管理対象接続は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる可能性があります。

デフォルトでは、このプロパティは設定されていません。

## ***XMSC\_WMQ\_SSL\_KEY\_RESETCOUNT***

**データ型:**

System.Int32

**プロパティ:**

ConnectionFactory

KeyResetCount は、秘密鍵の再ネゴシエーションが実行されるまで、1つの SSL 会話の中で送受信される暗号化されていないデータの合計バイト数を表します。このバイト数には、MCA によって送信される制御情報が含まれます。

XMS は、MQCONN を呼び出す前に、このプロパティに指定された値を MQSCO.KeyResetCount にコピーします。

パラメーター MQSCO.KeyRestCount は、IBM WebSphere MQ バージョン 6 からのみ使用可能です。IBM WebSphere MQ バージョン 5.3 の場合、このプロパティが設定されていると、XMS はキュー・マネージャーへの接続を試行せず、代わりに適切な例外をスローします。

ドット・ネットの場合のみ: IBM WebSphere MQ (WMQ\_CM\_CLIENT) への管理対象接続は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる可能性があります。

このプロパティのデフォルト値はゼロです。これは、秘密鍵が再ネゴシエーションされないことを意味します。

## **XMSC\_WM\_Q\_SSL\_PEER\_NAME**

**データ型:**  
     string

**プロパティ:**  
     ConnectionFactory

キュー・マネージャーとの SSL 接続で使用されるピア名。

このプロパティの標準値のリストはありません。その代わりに、SSLPEER の規則に従って、このstringを作成する必要があります。

ピア名の例を以下に示します。

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS は、MQCONN を呼び出す前に、このstringを正しい1バイト・コード・ページにコピーし、MQCD.SSLPeerNamePtr および MQCD.SSLPeerNameLength に正しい値を入れます。

アプリケーションがクライアント・モードでキュー・マネージャーに接続している場合のみ、このプロパティが関連します。

ドット・ネットの場合のみ: IBM WebSphere MQ への管理対象接続 (WMQ\_CM\_CLIENT) は SSL 接続をサポートしませんが、非管理対象接続 (WMQ\_CM\_CLIENT\_UNMANAGED) を使用することでサポートされる可能性があります。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WM\_Q\_SYNCPOINT\_ALL\_GETS**

**データ型:**  
     System.Boolean

**プロパティ:**  
     ConnectionFactory

すべてのメッセージを同期点制御の対象範囲内のキューから取り出す必要があるかどうかを示します。

プロパティの有効値は以下のとおりです。

<b>有効値</b>	<b>意味</b>
false	環境が適している場合、XMS クライアントは同期点制御の対象範囲外のキューからメッセージを取り出すことができます。
true	XMS クライアントは、すべてのメッセージを同期点制御の対象範囲内のキューから取り出す必要があります。

デフォルト値は false です。

## **XMSC\_WMQ\_TARGET\_CLIENT**

**データ型:**  
System.Int32

**プロパティ:**  
Destination

**URI で使用される名前:**  
targetClient

宛先に送信されるメッセージに MQRFH2 ヘッダーを付けるかどうかを示します。

アプリケーションが MQRFH2 ヘッダーのあるメッセージを送信する場合は、受信側のアプリケーションがこのヘッダーを処理できる必要があります。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WMQ_TARGET_DEST_JMS	宛先に送信されるメッセージには MQRFH2 ヘッダーがあります。アプリケーションが、MQRFH2 ヘッダーを処理する目的で設計されている別の XMS アプリケーション、WebSphere JMS アプリケーション、ネイティブの IBM WebSphere MQ アプリケーションのいずれかにメッセージを送信する場合は、この値を指定します。
XMSC_WMQ_TARGET_DEST_MQ	宛先に送信されるメッセージには MQRFH2 ヘッダーがありません。アプリケーションが、MQRFH2 ヘッダーを処理する目的で設計されているわけではないネイティブの IBM WebSphere MQ アプリケーションにメッセージを送信する場合は、この値を指定します。

デフォルト値は XMSC\_WMQ\_TARGET\_DEST\_JMS です。

## **XMSC\_WMQ\_TEMP\_Q\_PREFIX**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

アプリケーションが XMS 一時キューを作成するときに作成される IBM WebSphere MQ 動的キューの名前を形成するために使用される接頭部。

接頭部を形成するための規則は、オブジェクト記述子の **DynamicQName** フィールドの内容を形成するための規則と同じですが、最後の非空白文字はアスタリスク (\*) でなければなりません。このプロパティが設定されていない場合、使用される値は、z/OS では CSQ.\*、その他のプラットフォームでは AMQ.\* です。デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、Point-to-Point ・ドメインの場合に限られます。

## **XMSC\_WMQ\_TEMP\_TOPIC\_PREFIX**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory、Destination

一時トピックの作成時に、XMS は「TEMP/TEMPTOPICPREFIX/unique\_id」の形式のトピック・ストリングを生成します。このプロパティがデフォルト値のままである場合は、単に「TEMP/unique\_id」を生成します。空以外の値を指定すると、この接続で作成された一時トピックへのサブスクライバーの管理対象キューを作成するために、特定のモデル・キューを定義できます。

IBM WebSphere MQ トピックで有効な文字のみで構成されるヌル以外のストリングはすべて、このプロパティの有効な値です。

デフォルトでは、このプロパティは「」(空ストリング)に設定されています。

注: このプロパティが関連するのは、パブリッシュ/サブスクライブ・ドメインの場合に限られます。

### ***XMSC\_WMQ\_TEMPORARY\_MODEL***

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

アプリケーションが XMS 一時キューを作成するときに動的キューの作成元となる IBM WebSphere MQ モデル・キューの名前。

プロパティのデフォルト値は SYSTEM.DEFAULT.MODEL.QUEUE です。

このプロパティが関連するのは、Point-to-Point・ドメインの場合に限られます。

### ***XMSC\_WMQ\_WILDCARD\_FORMAT***

**データ型:**  
System.Int32

**プロパティ:**  
ConnectionFactory、Destination

このプロパティは、使用されるワイルドカード構文のバージョンを決定します。

IBM WebSphere MQ '\*' および '?' を指定してパブリッシュ/サブスクライブを使用する場合 ワイルドカードとして扱われます。一方、WebSphere メッセージ・ブローカーでパブリッシュ/サブスクライブを使用する場合は、「#」と「+」がワイルドカードとして扱われます。このプロパティは XMSC\_WMQ\_BROKER\_VERSION プロパティを置き換えます。

このプロパティの有効な値は以下のとおりです。

#### ***XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY***

トピック・レベルのワイルドカードのみを認識します。例: 「#」 および 「+」 はワイルドカードとして扱われます。この値は XMSC\_WMQ\_BROKER\_V2 と同じです。

#### ***XMSC\_WMQ\_WILDCARD\_CHAR\_ONLY***

文字ワイルドカードのみを認識します。例: '\*' および '?' ワイルドカードとして扱われます。この値は XMSC\_WMQ\_BROKER\_V1 と同じです。

デフォルトでは、このプロパティは XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY に設定されます。

注: IBM WebSphere MQ バージョン 6.0 以下を使用してパブリッシュ/サブスクライブを行う場合、このプロパティは関係ありません。代わりに、XMSC\_WMQ\_BROKER\_VERSION プロパティを使用する必要があります。

### ***XMSC\_WPM\_BUS\_NAME***

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory および Destination

**URI で使用される名前:**  
busName

接続ファクトリーの場合は、アプリケーションの接続先となるサービス統合バスの名前であり、宛先の場合は、その宛先が存在するサービス統合バスの名前です。

内容がトピックである宛先の場合、このプロパティは、関連するトピック・スペースが存在するサービス統合バスの名前になります。このトピック・スペースは、XMSC\_WPM\_TOPIC\_SPACE プロパティで指定します。

このプロパティを宛先に設定しなかった場合は、アプリケーションの接続先となるサービス統合バスにキューまたは関連するトピック・スペースが存在することが前提になります。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WPM\_CONNECTION\_PROTOCOL**

**データ型:**  
System.Int32

**プロパティ:**  
接続

メッセージング・エンジンへの接続に使用される通信プロトコルです。このプロパティは読み取り専用です。

このプロパティの可能な値は以下のとおりです。

値	意味
XMSC_WPM_CP_HTTP	接続には HTTP over TCP/IP を使用します。
XMSC_WPM_CP_TCP	接続には TCP/IP を使用します。

## **XMSC\_WPM\_CONNECTION\_PROXIMITY**

**データ型:**  
System.Int32

**プロパティ:**  
ConnectionFactory

接続に対する接続接近性の設定です。このプロパティを設定すると、アプリケーションが接続するメッセージング・エンジンがブートストラップ・サーバーにどの程度接近する必要があるかが指定されます。

プロパティの有効値は以下のとおりです。

有効値	接続接近性の設定
XMSC_WPM_CONNECTION_PROXIMITY_BUS	バス
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	クラスター
XMSC_WPM_CONNECTION_PROXIMITY_HOST	ホスト
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	サーバー

デフォルト値は XMSC\_WPM\_CONNECTION\_PROXIMITY\_BUS です。

## **XMSC\_WPM\_DUR\_SUB\_HOME**

**データ型:**  
ストリング

**プロパティ:**  
ConnectionFactory

**URI で使用される名前:**  
durableSubscriptionHome

ある接続または宛先に対するすべての永続サブスクリプションが管理対象になっているメッセージング・エンジンの名前です。永続サブスクリバラーに配信されるメッセージは、同じメッセージング・エンジンの発行点で保管されます。

接続を使用する永続サブスクリバラーをアプリケーションが作成する前に、その接続の永続サブスクリプション・ホームを指定する必要があります。宛先に対して指定されている値は、接続に対して指定されている値に優先します。

デフォルトでは、このプロパティは設定されていません。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

## ***XMSC\_WPM\_HOST\_NAME***

**データ型:**  
     文字列

**プロパティ:**  
     接続

アプリケーションの接続先となるメッセージング・エンジンが収容されているシステムのホスト名または IP アドレスです。このプロパティは読み取り専用です。

## ***XMSC\_WPM\_LOCAL\_ADDRESS***

**データ型:**  
     文字列

**プロパティ:**  
     ConnectionFactory

サービス統合バスへの接続の場合、このプロパティは、使用するローカル・ネットワーク・インターフェース、または使用するローカル・ポート (1 つまたは一定範囲)、あるいはその両方を指定します。

このプロパティの値は、次の形式の文字列です。

`[host_name][(low_port)[,high_port]]`

変数の意味は以下のとおりです。

### ***host\_name***

接続に使用するローカル・ネットワーク・インターフェースのホスト名または IP アドレスです。

この情報の入力が必要なのは、アプリケーションを実行しているシステムに複数のネットワーク・インターフェースがあり、接続にどのインターフェースを使用する必要があるかを指定できることが必要な場合に限ります。システムが備えるネットワーク・インターフェースが 1 つのみである場合、使用できるのはそのインターフェースのみです。システムに複数のネットワーク・インターフェースがあり、どのインターフェースを使用するかを指定していない場合、インターフェースはランダムに選択されます。

### ***low\_port***

接続に使用するローカル・ポートの数です。

*high\_port* も指定した場合、*low\_port* は一連のポート番号のうち最小のポート番号と解釈されます。

### ***high\_port***

一連のポート番号のうち最大のポート番号を表します。指定した範囲内のいずれかのポートを接続に使用する必要があります。

プロパティの有効値の例の一部を以下に示します。

```
JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)
```

デフォルトでは、このプロパティは設定されていません。

## ***XMSC\_WPM\_ME\_NAME***

**データ型:**  
    ストリング

**プロパティ:**  
    接続

アプリケーションの接続先となるメッセージング・エンジンの名前です。このプロパティは読み取り専用です。

## ***XMSC\_WPM\_NON\_PERSISTENT\_MAP***

**データ型:**  
    System.Int32

**プロパティ:**  
    ConnectionFactory

接続を使用して送信される非永続メッセージの信頼性レベルです。

プロパティの有効値は以下のとおりです。

### **有効値**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

デフォルト値は XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT です。

## ***XMSC\_WPM\_PERSISTENT\_MAP***

**データ型:**  
    System.Int32

**プロパティ:**  
    ConnectionFactory

接続を使用して送信される永続メッセージの信頼性レベルです。

プロパティの有効値は以下のとおりです。

### **有効値**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

### **信頼性レベル**

サービス統合バスでキューまたはトピック・スペースに指定されているデフォルトの信頼性レベルにより決定

ベスト・エフォート型の非永続

高速の非永続

高信頼性の非永続

高信頼性の永続

確実な永続

### **信頼性レベル**

サービス統合バスでキューまたはトピック・スペースに指定されているデフォルトの信頼性レベルにより決定

## 有効値

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

## 信頼性レベル

ベスト・エフォート型の非永続

高速の非永続

高信頼性の非永続

高信頼性の永続

確実な永続

デフォルト値は XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT です。

## **XMSC\_WPM\_PORT**

### データ型:

System.Int32

### プロパティ:

接続

アプリケーションの接続先となるメッセージング・エンジンによって listen されるポートの数です。このプロパティは読み取り専用です。

## **XMSC\_WPM\_PROVIDER\_ENDPOINTS**

### データ型:

ストリング

### プロパティ:

ConnectionFactory

ブートストラップ・サーバーの 1 つ以上のエンドポイント・アドレスの列です。エンドポイント・アドレスはコンマで区切ります。

ブートストラップ・サーバーとは、アプリケーションの接続先となるメッセージング・エンジンを選択する役割を果たすアプリケーション・サーバーのことです。ブートストラップ・サーバーのエンドポイント・アドレスの形式は次のとおりです。

*host\_name:port\_number:chain\_name*

エンドポイント・アドレスの構成要素の意味は、以下のとおりです。

### **host\_name**

ブートストラップ・サーバーが存在するシステムのホスト名または IP アドレスです。ホスト名または IP アドレスを指定していない場合、デフォルトは localhost になります。

### **port\_number**

ブートストラップ・サーバーが着信要求を listen するポートの数です。ポート番号を指定していない場合、デフォルトは 7276 になります。

### **chain\_name**

ブートストラップ・サーバーが使用するブートストラップ・トランスポート・チェーンの名前です。有効な値は以下のとおりです。

### 有効値

XMSC\_WPM\_BOOTSTRAP\_HTTP

XMSC\_WPM\_BOOTSTRAP\_HTTPS

ブートストラップ・トランスポート・チェーンの名前

BootstrapTunneledMessaging

BootstrapTunneledSecureMessaging

有効値	ブートストラップ・トランスポート・チェーンの名前
XMSC_WPM_BOOTSTRAP_SSL	BootstrapSecureMessaging
XMSC_WPM_BOOTSTRAP_TCP	BootstrapBasicMessaging

名前が指定されていない場合、デフォルト値は XMSC\_WPM\_BOOTSTRAP\_TCP です。

エンドポイント・アドレスを指定していない場合、デフォルトは localhost:7276:BootstrapBasicMessaging になります。

## **XMSC\_WPM\_TARGET\_GROUP**

**データ型:**  
 スtring

**プロパティ:**  
 ConnectionFactory

メッセージング・エンジンのターゲット・グループの名前です。ターゲット・グループの種類は、[XMSC\\_WPM\\_TARGET\\_TYPE](#) プロパティで決まります。

メッセージング・エンジンの検索対象を、サービス統合バス内のメッセージング・エンジンのサブグループに絞り込む場合には、このプロパティを設定してください。アプリケーションがサービス統合バス内の任意のメッセージング・エンジンに接続できるようにする場合は、このプロパティを設定しないでください。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WPM\_TARGET\_SIGNIFICANCE**

**データ型:**  
 System.Int32

**プロパティ:**  
 ConnectionFactory

メッセージング・エンジンのターゲット・グループの重要度です。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WPM_TARGET_SIGNIFICANCE_PREFERRED	ターゲット・グループ内にメッセージング・エンジンが存在する場合は、そのメッセージング・エンジンが選択されます。そうでない場合は、ターゲット・グループの外側に存在するメッセージング・エンジンが選択されます。ただし、そのメッセージング・エンジンが同じサービス統合バス内に存在することが前提です。
XMSC_WPM_TARGET_SIGNIFICANCE_REQUIRED	選択されたメッセージング・エンジンは、ターゲット・グループ内に存在する必要があります。ターゲット・グループ内にあるメッセージング・エンジンが使用不可の場合、該当する接続処理は失敗します。

プロパティのデフォルト値は XMSC\_WPM\_TARGET\_SIGNIFICANCE\_PREFERRED です。

## **XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN**

**データ型:**  
 String

### プロパティ:

ConnectionFactory

アプリケーションがメッセージング・エンジンに接続するために使用する必要があるインバウンド・トランスポート・チェーンの名前です。

このプロパティの値は、メッセージング・エンジンのホストとして動作するアプリケーション・サーバー内で使用できるいずれかのインバウンド・トランスポート・チェーンの名前です。事前に定義されているインバウンド・トランスポート・チェーンの1つには、次の名前付き定数が用意されています。

名前付き定数	トランスポート・チェーンの名前
XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC	InboundBasicMessaging

プロパティのデフォルト値は XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC です。

## **XMSC\_WPM\_TARGET\_TYPE**

### データ型:

System.Int32

### プロパティ:

ConnectionFactory

メッセージング・エンジンのターゲット・グループのタイプです。このプロパティは、XMSC\_WPM\_TARGET\_GROUP プロパティで指定されるターゲット・グループの種類を判別するプロパティです。

プロパティの有効値は以下のとおりです。

有効値	意味
XMSC_WPM_TARGET_TYPE_BUSMEMBER	ターゲット・グループの名前は、バス・メンバーの名前になります。このターゲット・グループの構成要素は、バス・メンバー内のすべてのメッセージング・エンジンです。
XMSC_WPM_TARGET_TYPE_CUSTOM	ターゲット・グループの名前は、メッセージング・エンジンのユーザー定義グループの名前になります。このターゲット・グループの構成要素は、ユーザー定義グループに登録されているすべてのメッセージング・エンジンです。
XMSC_WPM_TARGET_TYPE_ME	ターゲット・グループの名前は、メッセージング・エンジンの名前になります。ターゲット・グループは、指定されているメッセージング・エンジンです。

デフォルトでは、このプロパティは設定されていません。

## **XMSC\_WPM\_TEMP\_Q\_PREFIX**

### データ型:

ストリング

### プロパティ:

ConnectionFactory

アプリケーションが XMS 一時キューを作成するとサービス統合バスで作成される一時キューの名前を構成する場合に使用されるプレフィックスです。プレフィックスの最大長は 12 文字です。

一時キューの名前は文字「\_Q」で始まり、その後にプレフィックスが続きます。名前の残りの部分は、システム生成文字で構成されます。

デフォルトでは、このプロパティは設定されていません。つまり、一時キューの名前にプレフィックスはありません。

このプロパティが関連するのは、Point-to-Point・ドメインの場合に限られます。

### **XMSC\_WPM\_TEMP\_TOPIC\_PREFIX**

**データ型:**  
    ストリング

**プロパティ:**  
    ConnectionFactory

アプリケーションによって作成される一時トピックの名前を構成する場合に使用されるプレフィックスです。プレフィックスの最大長は 12 文字です。

一時トピックの名前は文字「\_T」で始まり、その後にプレフィックスが続きます。名前の残りの部分は、システム生成文字で構成されます。

デフォルトでは、このプロパティは設定されていません。つまり、一時トピックの名前にプレフィックスはありません。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。

### **XMSC\_WPM\_TOPIC\_SPACE**

**データ型:**  
    ストリング

**プロパティ:**  
    Destination

**URI で使用される名前:**  
    topicSpace

トピックが収容されているトピック・スペースの名前です。このプロパティを設定できるのは、内容がトピックである宛先のみです。

デフォルトでは、このプロパティは設定されていません。つまり、デフォルトのトピック・スペースが前提になります。

このプロパティが関連するのは、publish/subscribe・ドメインの場合に限られます。



## 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒 103-8510

東京都中央区日本橋箱崎町 19 番 21 号

日本アイ・ビー・エム株式会社

日本アイ・ビー・エム株式会社

法務・知的財産

U.S.A.

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

Intellectual Property Licensing

Legal and Intellectual Property Law

〒 103-8510

103-8510

東京 103-8510、日本

**以下の保証は、国または地域の法律に沿わない場合は、適用されません。** INTERNATIONAL BUSINESS MACHINES CORPORATION は、法律上の瑕疵担保責任、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。"" 国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

東京都中央区日本橋箱崎町 19 番 21 号

日本アイ・ビー・エム株式会社

Software Interoperability Coordinator, Department 49XA

3605 Highway 52 N

Rochester, MN 55901

U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名前はすべて架空のものであり、名前や住所が類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめかしたり、保証することはできません。

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

## プログラミング・インターフェース情報

プログラミング・インターフェース情報 (提供されている場合) は、このプログラムで使用するアプリケーション・ソフトウェアの作成を支援することを目的としています。

本書には、プログラムを作成するユーザーが IBM WebSphere MQ のサービスを使用するためのプログラミング・インターフェースに関する情報が記載されています。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

**重要:** この診断、修正、およびチューニング情報は、変更される可能性があるため、プログラミング・インターフェースとして使用しないでください。

## 商標

IBM、IBM ロゴ、ibm.com<sup>®</sup>は、世界の多くの国で登録された IBM Corporation の商標です。現時点での IBM の商標リストについては、"Copyright and trademark information" [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml) をご覧ください。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Linux® は、Linus Torvalds の米国およびその他の国における商標です。

この製品には、Eclipse Project (<http://www.eclipse.org/>) により開発されたソフトウェアが含まれています。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。







部品番号:

(1P) P/N: