8.0

# *IBM Message Service Client for .NET*

**참고**

이 정보와 이 정보가 지원하는 제품을 사용하기 전에, 231 페이지의 『주의사항』에 있는 정보를 확인하십시오.

이 개정판은 새 개정판에 별도로 명시하지 않는 한, IBM® MQ 의 버전 8릴리스 0및 모든 후속 릴리스와 수정에 적용됩니다.

IBM은 귀하가 IBM으로 보낸 정보를 귀하의 권리를 침해하지 않는 범위 내에서 적절하다고 생각하는 방식으로 사용하거나 배포할 수 있습니다.

# 목차

# IBM Message Service Client for .NET 소개

IBM Message Service Client for .NET는 Java™ Message Service(JMS) API와 동일한 인터페이스 세트가 있는 XMS라는 API(Application Programming Interface)를 제공합니다. IBM Message Service Client for .NET에는 XMS의 전체 관리 구현이 포함되며, 이는 .NET를 준수하는 어에서 사용할 수 있습니다.

XMS에서는 다음을 지원합니다.

- 포인트-투-포인트 메시징
- 발행/구독 메시징
- 동기 메시지 전달
- 비동기 메시지 전달

XMS 애플리케이션은 다음 유형의 애플리케이션과 메시지를 교환할 수 있습니다.

- XMS 애플리케이션
- IBM MQ classes for JMS 애플리케이션
- 기본 IBM MQ 애플리케이션
- IBM MQ 기본 메시징 제공자를 사용하는 JMS 애플리케이션

XMS 애플리케이션은 다음 메시징 서버에 연결하고 이에 대한 자원을 사용할 수 있습니다.

**IBM MQ 큐 관리자**
애플리케이션은 바인딩 또는 클라이언트 모드에서 연결될 수 있습니다.

**WebSphere® Application Server 서비스 통합 버스**
애플리케이션은 직접 TCP/IP 연결을 사용하거나 TCP/IP를 통한 HTTP를 사용할 수도 있습니다.

**IBM Integration Bus**
메시지는 WebSphere MQ Real-time Transport를 사용하여 애플리케이션과 브로커 사이에서 전송됩니다. 메시지는 WebSphere MQ 멀티캐스트 전송를 사용하여 애플리케이션에 전달될 수 있습니다.

IBM MQ 큐 관리자에 연결하면 XMS 애플리케이션이 WebSphere MQ Enterprise Transport를 사용하여 IBM Integration Bus와 통신할 수 있습니다. 또는 XMS 애플리케이션이 WebSphere MQ에 연결하여 발행 및 구독할 수 있습니다.

**관련 개념**
6 페이지의 『메시징 스타일』
7 페이지의 『XMS 오브젝트 모델』
XMS API는 오브젝트 지향 인터페이스입니다. XMS 오브젝트 모델은 JMS 1.1 오브젝트 모델을 기반으로 합니다.

9 페이지의 『XMS 메시지 모델』
XMS 메시지 모델은 JMS용 WebSphere MQ 클래스 메시지 모델과 같습니다.

# IBM Message Service Client for .NET 소개

IBM Message Service Client for .NET는 Java Message Service(JMS) API와 동일한 인터페이스 세트가 있는 XMS라는 API(Application Programming Interface)를 제공합니다. IBM Message Service Client for .NET에는 XMS의 전체 관리 구현이 포함되며, 이는 .NET를 준수하는 어에서 사용할 수 있습니다.

XMS에서는 다음을 지원합니다.

- 포인트-투-포인트 메시징
- 발행/구독 메시징
- 동기 메시지 전달
- 비동기 메시지 전달

XMS 애플리케이션은 다음 유형의 애플리케이션과 메시지를 교환할 수 있습니다.

- XMS 애플리케이션
- IBM MQ classes for JMS 애플리케이션
- 기본 IBM MQ 애플리케이션
- IBM MQ 기본 메시징 제공자를 사용하는 JMS 애플리케이션

XMS 애플리케이션은 다음 메시징 서버에 연결하고 이에 대한 자원을 사용할 수 있습니다.

**IBM MQ 큐 관리자**
애플리케이션은 바인딩 또는 클라이언트 모드에서 연결될 수 있습니다.

**WebSphere Application Server 서비스 통합 버스**
애플리케이션은 직접 TCP/IP 연결을 사용하거나 TCP/IP를 통한 HTTP를 사용할 수도 있습니다.

**IBM Integration Bus**
메시지는 WebSphere MQ Real-time Transport를 사용하여 애플리케이션과 브로커 사이에서 전송됩니다.
메시지는 WebSphere MQ 멀티캐스트 전송를 사용하여 애플리케이션에 전달될 수 있습니다.

IBM MQ 큐 관리자에 연결하면 XMS 애플리케이션이 WebSphere MQ Enterprise Transport를 사용하여 IBM Integration Bus와 통신할 수 있습니다. 또는 XMS 애플리케이션이 WebSphere MQ에 연결하여 발행 및 구독할 수 있습니다.

**관련 개념**
6 페이지의 『메시징 스타일』
7 페이지의 『XMS 오브젝트 모델』
XMS API는 오브젝트 지향 인터페이스입니다. XMS 오브젝트 모델은 JMS 1.1 오브젝트 모델을 기반으로 합니다.

9 페이지의 『XMS 메시지 모델』
XMS 메시지 모델은 JMS용 WebSphere MQ 클래스 메시지 모델과 같습니다.

# 메시징 스타일

XMS는 메시징의 포인트-투-포인트 및 발행/구독 스타일을 지원합니다.

메시징 스타일을 메시징 도메인이라고도 합니다.

## 포인트-투-포인트 메시징

포인트-투-포인트 메시징의 일반 양식은 큐를 사용합니다. 가장 간단한 경우 애플리케이션은 목적지 큐를 암시적 또는 명시적으로 식별하여 다른 애플리케이션으로 메시지를 전송합니다. 기본 메시징 및 큐잉 시스템은 전송 애플리케이션으로부터 메시지를 수신하고 메시지를 목적지 큐로 라우팅합니다. 그런 다음 수신 애플리케이션이 큐에서 메시지를 검색합니다.

기본 메시징 및 큐잉 시스템에 IBM Integration Bus가 포함되는 경우 IBM Integration Bus는 메시지를 복제하고 메시지 사본을 다른 큐로 라우팅할 수 있습니다. 그러면 둘 이상의 애플리케이션이 메시지를 수신할 수 있습니다. IBM Integration Bus는 메시지를 변환하고 데이터를 여기에 추가할 수도 있습니다.

포인트-투-포인트 메시징의 주요 특징은 메시지를 송신할 때 애플리케이션이 로컬 큐에 메시지를 놓는다는 것입니다. 기본 메시징 및 큐잉 시스템은 메시지를 전송할 목적지 큐를 판별합니다. 기본 애플리케이션은 목적지 큐로부터 메시지를 수신합니다.

## 발행/구독 메시징

발행/구독 메시징에는 발행자와 구독자, 두 유형의 애플리케이션이 있습니다.

발행자는 발행 메시지 양식으로 정보를 제공합니다. 발행자가 메시지를 발행하면, 이는 메시지 내에서 정보의 토픽을 식별하는 토픽을 지정합니다.

구독자는 발행된 정보의 이용자입니다. 구독자는 구독을 작성하여 관심있는 토픽을 지정합니다.

발행/구독 시스템은 발행자의 발행물 그리고 구독자의 구독을 수신합니다. 이는 구독자에게 발행물을 라우팅합니다. 구독자는 구독하는 해당 토픽에 대한 발행물을 수신합니다.

발행/구독 메시징의 주요 특징은 발행자가 메시지를 발행할 때 발행자가 주제를 식별하는 것입니다. 이는 구독자를 식별하지 않습니다. 메시지가 구독자가 없는 토픽에 대해 발행되면 애플리케이션은 메시지를 수신하지 않습니다.

애플리케이션은 발행자 및 구독자 모두가 될 수 있습니다.

## XMS 오브젝트 모델

XMS API는 오브젝트 지향 인터페이스입니다. XMS 오브젝트 모델은 JMS 1.1 오브젝트 모델을 기반으로 합니다.

다음 목록은 기본 XMS 클래스 또는 오브젝트 유형을 요약합니다.

**ConnectionFactory**
ConnectionFactory 오브젝트는 연결의 매개변수 세트를 캡슐화합니다. 애플리케이션은 ConnectionFactory를 사용하여 연결을 작성합니다. 애플리케이션은 런타임 시 매개변수를 제공하고 ConnectionFactory 오브젝트를 작성할 수 있습니다. 또는 연결 매개변수가 관리 대상 오브젝트 저장소에 저장될 수 있습니다. 애플리케이션은 저장소에서 오브젝트를 검색하고 여기에서 ConnectionFactory 오브젝트를 작성할 수 있습니다.

**Connection**
Connection 오브젝트는 애플리케이션에서 메시징 서버로서의 활성 연결을 캡슐화합니다. 애플리케이션은 연결을 사용하여 세션을 작성합니다.

**Destination**
애플리케이션은 Destination 오브젝트를 사용하여 메시지를 전송하고 메시지를 수신합니다. 발행/구독 도메인에서 Destination 오브젝트는 토픽을 캡슐화하고 포인트-투-포인트 도메인에서는 Destination 오브젝트가 큐를 캡슐화합니다. 애플리케이션은 런타임 시 Destination 오브젝트를 작성하는 매개변수를 제공할 수 있습니다. 또는 관리 대상 오브젝트 저장소에 저장되는 오브젝트 정의에서 Destination 오브젝트를 작성할 수 있습니다.

**Session**
Session 오브젝트는 메시지 전송 및 수신을 위한 단일 스레드 컨텍스트입니다. 애플리케이션은 Session 오브젝트를 사용하여 Message, MessageProducer 및 MessageConsumer 오브젝트를 작성합니다.

**Message**
Message 오브젝트는 애플리케이션이 MessageProducer 오브젝트를 사용하여 전송하거나 MessageConsumer 오브젝트를 사용하여 수신하는 Message 오브젝트를 캡슐화합니다.

**MessageProducer**
MessageProducer 오브젝트는 애플리케이션이 목적지로 메시지를 전송할 때 사용됩니다.

**MessageConsumer**
MessageConsumer 오브젝트는 애플리케이션이 목적지로 전송된 메시지를 수신할 때 사용됩니다.

8 페이지의 그림 1에서는 이러한 오브젝트와 해당 관계를 표시합니다. 이 다이어그램은 XMS 오브젝트의 기본 유형, 즉 ConnectionFactory, Connection, Session, MessageProducer, MessageConsumer, Message 및 Destination을 보여줍니다. 애플리케이션은 연결 팩토리를 사용하여 연결을 작성하고 연결을 사용하여 세션을 작성합니다. 그런 다음 애플리케이션은 세션을 사용하여 메시지, 메시지 생성자 및 메시지 이용자를 작성할 수 있습니다. 애플리케이션은 메시지 생성자를 사용하여 메시지를 목적지로 송신하고 메시지 이용자를 사용하여 목적지로 송신된 메시지를 수신합니다.

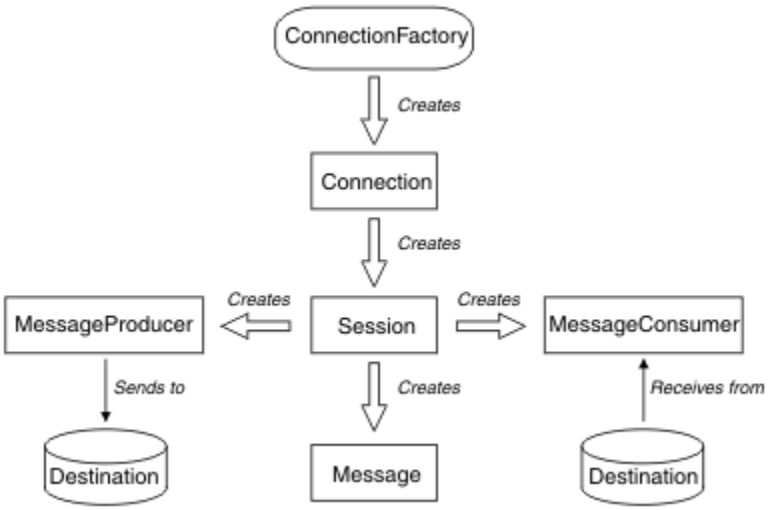


그림 *1. XMS* 오브젝트 및 해당 관계

.NET에서 XMS 클래스는 .NET 인터페이스 세트로 정의됩니다. XMS .NET 애플리케이션을 코드화하는 경우, 선언된 인터페이스만 필요합니다.

XMS 오브젝트 모델은 *Java Message Service Specification,* 버전 *1.1*에 설명된 도메인 독립적인 인터페이스를 기반으로 합니다. `Topic`, `TopicPublisher` 및 `TopicSubscriber`와 같은 도메인 특정 클래스는 제공되지 않습니다.

## 오브젝트의 속성 및 특성

XMS 오브젝트에는 여러 가지 방법으로 구현되는 오브젝트 특성인 속성과 특성을 포함할 수 있습니다.

**속성**
속성에 값이 없는 경우에도 항상 제공되며 스토리지를 사용하는 오브젝트 특성입니다. 이 점에서 속성은 고정 길이 데이터 구조의 필드와 유사합니다. 속성의 고유한 특징은 각 속성에 해당 값을 설정하고 가져오는 데 필요한 자체 메소드가 있다는 점입니다.

**특성**
값이 설정되어야 오브젝트의 특성이 제공되고 스토리지를 사용합니다. 값을 설정한 후 특성을 삭제하거나 스토리지를 복구할 수 없습니다. 값은 변경할 수 있습니다. XMS에서는 특성 값을 설정 및 가져오기 위한 일반 메소드 세트를 제공합니다.

**관련 개념**
XMS 기본 유형
XMS에서는 동일한 8개의 Java 기본 유형(byte, short, int, long, float, double, char 및 boolean)을 제공합니다. 이를 사용하여 데이터 손실이나 손상 없이 XMS와 JMS 간에 메시지를 교환할 수 있습니다.

한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환
애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

**관련 참조**
애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

## 관리 대상 오브젝트

관리 대상 오브젝트를 사용하여 중앙 저장소에서 관리 대상 클라이언트 애플리케이션이 사용하는 연결 설정을 관리할 수 있습니다. 애플리케이션은 중앙 저장소에서 오브젝트 정의를 검색하고 이를 사용하여 `ConnectionFactory` 및 `Destination` 오브젝트를 작성합니다. 관리 대상 오브젝트를 사용하면 런타임 시 사용하는 자원에서 애플리케이션을 커플링 해제할 수 있습니다.

예를 들면 테스트 환경에서 XMS 애플리케이션을 작성하여, 목적지 및 연결 세트를 참조하는 관리 대상 오브젝트와 함께 테스트할 수 있습니다. 애플리케이션이 배치되면 관리 대상 오브젝트는 프로덕션 환경에서 연결 및 목적지를 참조하는 애플리케이션을 구성하도록 변경될 수 있습니다.

XMS는 두 가지 유형의 관리 대상 오브젝트를 지원합니다.

- 애플리케이션이 서버에 대한 초기 연결을 작성하는 데 사용하는 `ConnectionFactory` 오브젝트.
- `Destination` 오브젝트는 애플리케이션이 전송되는 메시지의 목적지 및 수신되는 메시지의 소스를 지정할 때 사용합니다. 목적지는 애플리케이션이 연결하는 서버의 토픽이나 큐입니다.

관리 도구 **JMSAdmin**은 WebSphere MQ와 함께 제공됩니다. 관리 오브젝트의 중앙 저장소에서 에 대한 관리 오브젝트를 작성하고 관리하는 데 사용됩니다.

저장소에 있는 관리 대상 오브젝트는 JMS용 WebSphere MQ 클래스 및 XMS 애플리케이션이 사용할 수 있습니다. XMS applications can use the `ConnectionFactory` and `Destination` objects to connect to a WebSphere MQ 큐 관리자. 관리자는 애플리케이션 코드에 영향을 주지 않고 저장소에 보관된 오브젝트 정의를 변경할 수 있습니다.

다음 다이어그램은 XMS 애플리케이션에서 관리 대상 오브젝트를 사용하는 일반적인 방법을 보여줍니다. 다이어그램의 왼쪽에서는 관리 콘솔을 사용하여 관리하는 ConnectionFactory 및 Destination 오브젝트 정의가 포함되는 저장소를 표시합니다. 다이어그램의 오른쪽에는 저장소에서 오브젝트 정의를 찾아본 후, 메시징 서버에 연결할 때 이 오브젝트 정의를 사용하는 XMS 애플리케이션이 표시됩니다.

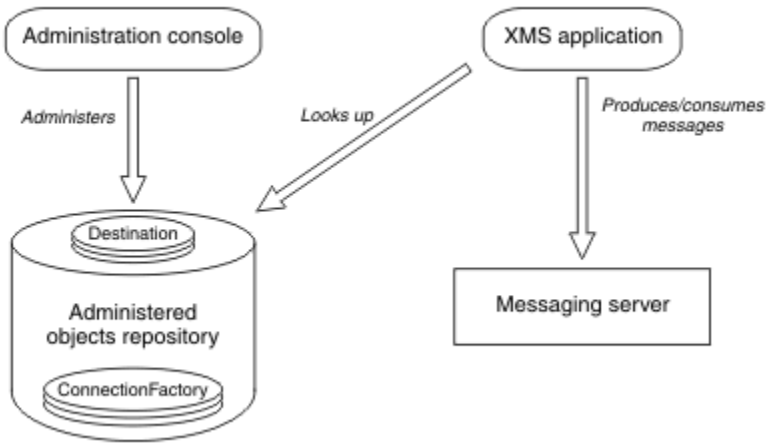


그림 *2. XMS* 애플리케이션이 일반적으로 관리 대상 오브젝트를 사용하는 방법

**관련 개념**

관리 대상 오브젝트에 대한 작업
이 장에서는 관리 오브젝트에 대한 정보를 제공합니다. XMS 애플리케이션은 중앙 관리 대상 오브젝트 저장소에서 오브젝트 정의를 검색하여 연결 팩토리와 목적지를 작성하는 데 사용할 수 있습니다.

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

## XMS 메시지 모델

XMS 메시지 모델은 JMS용 WebSphere MQ 클래스 메시지 모델과 같습니다.

특히 XMS는 JMS용 WebSphere MQ 클래스가 구현하는 것과 동일한 메시지 특성과 메시지 헤더 필드를 구현합니다.

- JMS 헤더 필드. 이러한 필드의 이름은 접두부 JMS로 시작합니다.
- JMS 정의 특성. 이러한 필드에는 이름이 접두부 JMSX로 시작하는 특성이 있습니다.
- IBM 정의 특성. 이러한 필드에는 이름이 접두부 JMS_IBM_로 시작하는 특성이 있습니다.

그러므로 XMS 애플리케이션은 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있습니다. 각 메시지에서 일부 헤더 필드 및 특성은 애플리케이션에서 설정하고 나머지는 XMS 또는 JMS용 WebSphere MQ 클래스에 의해 설정됩니다. XMS 또는 JMS용 WebSphere MQ 클래스에서 설정하는 필드 일부는 메시지를 전송할 때 설정되고, 기타는 수신할 때 설정됩니다. 헤더 필드 및 특성은 적절한 경우 메시징 서버를 통해 메시지와 함께 전파됩니다. 이는 메시지를 수신하는 애플리케이션에서 사용할 수 있습니다.

# 애플리케이션에서 새 XMS 버전 사용 방지

기본적으로, 최신 XMS 버전을 설치하는 경우, 재컴파일할 필요 없이 이전 버전을 사용하는 애플리케이션이 최신 버전으로 자동 전환됩니다.

## 이 태스크 정보

다중 버전 공존 기능을 사용하면 신규 XMS 버전이 설치되어도 이전 XMS 버전을 덮어쓰지 않습니다. 대신 유사한 XMS .NET의 여러 인스턴스 어심블리들이 GAC(Global Assembly Cache)에 공존하며 이들의 버전 번호는 다릅니다. 내부적으로 GAC는 정책 파일을 사용하여 애플리케이션 호출을 최신 버전의 XMS로 라우팅합니다. 다시 컴파일할 필요 없이 애플리케이션이 실행되고 신규 XMS .NET 버전에서 새 기능을 사용할 수 있습니다.

그러나 이전 XMS 버전을 사용하는 데 애플리케이션이 필요한 경우 애플리케이션 구성 파일에서 `publisherpolicy` 속성을 no 로 설정하십시오.

**참고:** 애플리케이션 구성 파일은 접미어가 .config이고 파일이 관련되는 실행 프로그램 이름으로 구성되는 파일입니다. 예를 들어 text.exe에 대한 애플리케이션 구성 파일의 이름은 text.exe.config가 됩니다.

그러나 언제든지 시스템의 모든 애플리케이션은 동일한 버전의 XMS .NET를 사용합니다.

# 메시징 서버 환경 설정

이 장에서는 XMS 애플리케이션이 서버에 연결할 수 있도록 메시징 서버 환경을 설정하는 방법에 대해 설명합니다.

WebSphere MQ 큐 관리자에 연결하는 애플리케이션의 경우에는 WebSphere MQ 클라이언트(또는 바인딩 모드의 큐 관리자)가 필요합니다.

현재, 브로커에 대한 실시간 연결을 사용하는 애플리케이션의 경우 전제조건이 없습니다.

XMS에서 제공하는 샘플 애플리케이션을 포함하여 XMS 애플리케이션을 실행하기 전에 메시징 서버 환경을 설정해야 합니다.

이 장에는 다음 절가 포함됩니다.


- 11 페이지의 『WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성』
- 12 페이지의 『브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성』
- 13 페이지의 『WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성』


**관련 개념**

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

XMS 샘플 애플리케이션 사용
XMS와 함께 제공된 샘플 애플리케이션을 사용하여 설치 및 메시징 서버 설정을 확인하고 고유한 애플리케이션을 빌드하도록 도와줍니다. 샘플은 각 API의 공통 기능에 대해 간략하게 설명합니다.

**관련 태스크**

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

**관련 참조**

WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건
XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

# WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성

이 섹션은 WebSphere MQ 버전 7.0 이상을 사용하고 있다고 가정합니다. WebSphere MQ 큐 관리자에 연결되는 애플리케이션을 실행하기 전에 큐 관리자를 구성해야 합니다. 발행/구독 애플리케이션의 경우 큐에 있는 발행/구독 인터페이스를 사용하는 경우 일부 추가 구성이 필요합니다.

## 시작하기 전에

XMS는 IBM Integration Bus 또는 WebSphere Message Broker 버전 6.1 이상과 함께 작동합니다.

이 태스크를 시작하기 전에 다음 단계를 수행하십시오.

- 애플리케이션에 실행 중인 큐 관리자에 액세스 권한이 있는지 확인하십시오.
- 애플리케이션이 발행/구독 애플리케이션이고 큐에 있는 발행/구독 인터페이스를 사용하는 경우, "PSMODE" 속성이 큐 관리자에 "ENABLED"으로 설정되었는지 확인하십시오.
- 애플리케이션이 큐 관리자에 연결하도록 특성이 적절히 설정되어 있는 연결 팩토리를 사용하는지 확인하십시오. 애플리케이션이 발행/구독 애플리케이션인 경우, 브로커 사용에 적합하도록 애플리케이션 연결 팩토리 특성이 설정되어 있는지 확인하십시오. 연결 팩토리의 특성에 대한 자세한 정보는 169 페이지의 『ConnectionFactory의 특성』의 내용을 참조하십시오.

## 이 태스크 정보

WebSphere MQ JMS 애플리케이션을 실행하도록 큐 관리자와 큐에 있는 발행/구독 인터페이스를 구성하는 방식과 동일한 방식으로 XMS 애플리케이션을 실행하도록 큐 관리자 및 브로커를 구성합니다. 다음은 수행해야 하는 작업을 요약한 단계입니다.

## 프로시저

1. 큐 관리자에서 애플리케이션에 필요한 큐를 작성하십시오.

   큐를 작성하는 방법에 대한 개요는 큐 정의를 참조하십시오.

   애플리케이션이 발행/구독 애플리케이션이고 JMS용 WebSphere MQ 클래스 시스템 큐에 액세스해야 하는 큐에 있는 발행/구독 인터페이스를 사용하는 경우, 큐를 작성하기 전에 4a 단계까지 기다리십시오.
2. 애플리케이션과 연관된 사용자 ID에 큐 관리자에 연결할 수 있는 권한과 해당 큐에 액세스할 수 있는 적절한 권한을 부여하십시오.

   권한 부여에 대한 개요는 보안을 참조하십시오. 애플리케이션이 클라이언트 모드에서 큐 관리자에 연결되는 경우, 클라이언트 및 서버도 참조하십시오.
3. 애플리케이션이 클라이언트 모드에서 큐 관리자에 연결되는 경우, 서버 연결 채널이 큐 관리자에 정의되고 리스너가 시작되었는지 확인하십시오.

   큐 관리자에 연결되는 각 애플리케이션에서 이 단계를 수행할 필요가 없습니다. 하나의 서버 연결 채널 정의와 하나의 리스너가 클라이언트 모드에서 연결되는 모든 애플리케이션을 지원할 수 있습니다.
4. 애플리케이션이 발행/구독 애플리케이션이고 큐에 있는 발행/구독 인터페이스를 사용하는 경우, 다음 단계를 수행하십시오.

   a) 큐 관리자에서 WebSphere MQ와 함께 제공되는 MQSC 명령의 스크립트를 실행하여 JMS용 WebSphere MQ 클래스 시스템 큐를 작성하십시오. WebSphere Message Broker와 연관된 사용자 ID에 큐에 액세스할 권한이 있는지 확인하십시오.

      스크립트 찾는 위치 및 실행 방법에 대한 정보는 Java 사용의 내용을 참조하십시오.

큐 관리자에 대해 이 단계를 한 번만 수행하십시오. 동일한 JMS용 WebSphere MQ 클래스 시스템 큐 세트는 동일한 세트는 큐 관리자에 연결되는 모든 XMS 및 JMS용 WebSphere MQ 클래스 애플리케이션을 지원할 수 있습니다.

b) 애플리케이션과 연관되는 사용자 ID에 JMS용 WebSphere MQ 클래스 시스템 큐에 액세스하는 권한을 부여하십시오.

사용자 ID에게 필요한 권한 정보는 JMS 사용하기을 참조하십시오.

c) IBM Integration Bus의 브로커를 사용하려면, 메시지 플로우를 작성하고 배치하여 애플리케이션이 발행하는 메시지를 송신하는 큐를 제공하십시오.

기본 메시지 플로우는 발행된 메시지를 읽기 위한 MQInput 메시지 처리 노드와 메시지를 발행하기 위한 Publication 메시지 처리 노드를 포함합니다.

메시지 플로우를 작성하고 배치하는 방법에 대한 정보는 IBM Integration Bus 제품 문서를 참조하십시오.

적합한 메시지 플로우가 이미 브로커에 배치된 경우 이러한 단계를 수행할 필요가 없습니다.

### 결과

이제 애플리케이션을 시작할 수 있습니다.

**관련 태스크**

브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성
브로커에 대한 실시간 연결을 사용하는 애플리케이션을 실행하려면 해당 브로커를 구성해야 합니다.

WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성
WebSphere 서비스 통합 버스에 연결되는 애플리케이션을 실행하기 전에, 기본 메시징 제공자를 사용하는 JMS 애플리케이션을 실행하도록 서비스 통합 버스를 구성하는 방식과 동일한 방법으로 서비스 통합 버스를 구성해야 합니다.

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

**관련 참조**

WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건
XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

## 브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성

브로커에 대한 실시간 연결을 사용하는 애플리케이션을 실행하려면 해당 브로커를 구성해야 합니다.

### 시작하기 전에

이 태스크를 시작하기 전에 다음 단계를 수행하십시오.

- 애플리케이션이 실행 중인 브로커에 액세스 권한을 가지고 있는지 확인하십시오.
- 애플리케이션이 브로커에 대한 실시간 연결에 적합하도록 특성이 설정되어 있는 연결 팩토리를 사용하는지 확인하십시오. 연결 팩토리의 특성에 대한 자세한 정보는 169 페이지의 『ConnectionFactory의 특성』의 내용을 참조하십시오.

### 이 태스크 정보

XMS 애플리케이션을 실행하도록 브로커를 구성하는 것과 동일한 방식으로 JMS용 WebSphere MQ 클래스 애플리케이션을 실행하도록 브로커를 구성하십시오. 다음은 수행해야 할 작업을 요약한 단계입니다.

### 프로시저

1. 브로커가 청취하고 메시지를 발행하는 TCP/IP에서 메시지를 읽는 메시지 플로우를 작성 및 배치합니다.

   다음 방법 중 하나로 이를 수행할 수 있습니다.

   - **Real-timeOptimizedFlow** 메시지 처리 노드를 포함하는 메시지 플로우를 작성합니다.

- **Real-timeInput** 메시지 처리 노드 및 Publication 메시지 처리 노드를 포함하는 메시지 플로우를 작성합니다.

실시간 연결에 사용되는 포트를 청취하도록 **Real-timeOptimizedFlow** 또는 **Real-timeInput** 노드를 구성해야 합니다. XMS에서 실시간 연결을 위한 기본 포트 번호는 1506입니다.

적합한 메시지 플로우가 이미 브로커에 배치된 경우 이러한 단계를 수행할 필요가 없습니다.

2. WebSphere MQ 멀티캐스트 전송를 사용하여 애플리케이션에 메시지를 전달해야 할 경우 멀티캐스트가 가능하도록 브로커를 구성하십시오. 신뢰할 수 있는 멀티캐스트가 필요한 해당 토픽의 신뢰 가능한 서비스 품질(QoS)을 지정하여 멀티캐스트가 가능해야 하는 토픽을 구성하십시오.
3. 브로커에 연결할 때 애플리케이션이 사용자 ID와 비밀번호를 제공하고 이러한 정보를 이용하여 브로커가 애플리케이션을 인증할 수 있도록 하려면, 단순한 텔넷과 유사한 비밀번호 인증에 적합하도록 사용자 이름 서버와 브로커를 구성하십시오.

### 결과

이제 애플리케이션을 시작할 수 있습니다.

**관련 태스크**

WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성
이 섹션은 WebSphere MQ 버전 7.0 이상을 사용하고 있다고 가정합니다. WebSphere MQ 큐 관리자에 연결되는 애플리케이션을 실행하기 전에 큐 관리자를 구성해야 합니다. 발행/구독 애플리케이션의 경우 큐에 있는 발행/구독 인터페이스를 사용하는 경우 일부 추가 구성이 필요합니다.

WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성
WebSphere 서비스 통합 버스에 연결되는 애플리케이션을 실행하기 전에, 기본 메시징 제공자를 사용하는 JMS 애플리케이션을 실행하도록 서비스 통합 버스를 구성하는 방식과 동일한 방법으로 서비스 통합 버스를 구성해야 합니다.

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

**관련 참조**

WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건
XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

## WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성

WebSphere 서비스 통합 버스에 연결되는 애플리케이션을 실행하기 전에, 기본 메시징 제공자를 사용하는 JMS 애플리케이션을 실행하도록 서비스 통합 버스를 구성하는 방식과 동일한 방법으로 서비스 통합 버스를 구성해야 합니다.

### 시작하기 전에

이 태스크를 시작하기 전에 다음 단계를 수행해야 합니다.

- 메시징 버스가 작성되고 서버가 버스 멤버로서 버스에 추가되는지 확인하십시오.
- 애플리케이션이 실행 중인 하나 이상의 메시징 엔진을 포함하는 서비스 통합 버스에 액세스할 수 있는지 확인합니다.
- HTTP 조작이 필요할 경우 HTTP 메시징 엔진 인바운드 전송 채널을 정의해야 합니다. 기본적으로 SSL 및 TCP에 대한 채널은 서버 설치 도중 정의됩니다.
- 애플리케이션이 부트스트랩 서버를 사용하여 서비스 통합 버스에 연결되도록 적절하게 특성이 설정된 연결 팩토리를 사용하는지 확인합니다. 필요한 최소 정보는 다음과 같습니다.
  - 메시징 서버로의 연결을 조정할 때 사용하는 위치와 프로토콜에 대해 설명하는 제공자 엔드포인트입니다(즉, 부트스트랩 서버를 통해 조정). 가장 단순한 형식으로, 기본 설정으로 설정된 서버의 경우, 제공자 엔드포인트를 서버의 호스트 이름으로 설정할 수 있습니다.
  - 메시지를 전송하는 버스 이름.

  연결 팩토리의 특성에 대한 자세한 정보는 169 페이지의 『ConnectionFactory의 특성』의 내용을 참조하십시오.

### 이 태스크 정보

필요한 큐 또는 토픽 공간을 정의해야 합니다. 기본적으로 Default.Topic.Space 토픽 공간이 서버 설치 도중 정의되지만 추가 토픽 공간이 필요한 경우 스스로 이러한 토픽 공간을 작성해야 합니다. 서버는 필요에 따라 동적으로 이러한 개별 토픽을 인스턴스화하므로 토픽 공간 내에서 개별 토픽을 사전 정의할 필요는 없습니다.

다음은 수행해야 하는 작업을 요약한 단계입니다.

### 프로시저

1. 포인트-투-포인트 메시징을 위해 애플리케이션에 필요한 큐를 작성하십시오.
2. 발행/구독 메시징을 위해 애플리케이션에 필요한 추가 토픽 영역을 작성하십시오.

### 결과

이제 애플리케이션을 시작할 수 있습니다.

**관련 태스크**

WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성
이 섹션은 WebSphere MQ 버전 7.0 이상을 사용하고 있다고 가정합니다. WebSphere MQ 큐 관리자에 연결되는 애플리케이션을 실행하기 전에 큐 관리자를 구성해야 합니다. 발행/구독 애플리케이션의 경우 큐에 있는 발행/구독 인터페이스를 사용하는 경우 일부 추가 구성이 필요합니다.

브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성
브로커에 대한 실시간 연결을 사용하는 애플리케이션을 실행하려면 해당 브로커를 구성해야 합니다.

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

**관련 참조**

WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건
XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

## 설치 마법사를 사용하여 Message Service Client for .NET 설치

설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

### 이 태스크 정보

Windows에 Message Service Client for .NET를 설치하려면 다음 프로시저를 수행하십시오.

### 프로시저

1. SupportPac에서 설치하는 경우 다음 단계를 완료하거나 14 페이지의 『2』 단계로 바로 이동하십시오.
   a) 윈도우에서 관리자로 로그온하십시오.
   b) dotNETClientsetup.exe 설치 프로그램을 실행하십시오.
2. 설치 마법사가 표시되고 다음 메시지가 표시될 때까지 기다리십시오.

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

   **다음**을 클릭하십시오.

   마법사에서 라이센스 계약을 읽도록 요청합니다.
3. 라이센스 계약을 읽도록 요청하고 라이센스 계약의 조항에 동의하는 경우, **라이센스 계약의 조항에 동의합니다**를 클릭한 후 **다음**을 클릭하십시오.

   설치 마법사가 사용자 요구사항에 가장 잘 맞는 설치 유형을 선택하도록 요청합니다.
4. 필요한 설치 유형을 선택하십시오.

- 모든 프로그램 기능을 설치하고 이를 기본 설치 디렉토리에 설치하려면 **전체**를 클릭하십시오.
- 설치하려는 기능을 선택하고 설치 위치를 지정하려면 **사용자 정의**를 지정하십시오.

5. **다음**을 클릭하십시오.

   전체 설치 옵션을 선택하는 경우, 15 페이지의 『8』 단계에 설명된 대로 설치 마법사에 설치를 시작할 준비가 되었음을 나타내는 메시지가 표시됩니다. 사용자 정의 설치 옵션을 선택하는 경우, 설치 마법사에서 설치하려는 기능을 선택하고 15 페이지의 『6』 단계와 15 페이지의 『7』 단계를 완료한 후 15 페이지의 『8』 단계로 이동해야 합니다.
6. 사용자 정의 설치의 경우에는, Message Service Client for .NET 기능이 설치되는 방식에 대한 변경 사항을 지정하려면 기능 목록의 아이콘을 클릭하십시오. 제안된 디렉토리에 Message Service Client for .NET를 설치하지 않으려면 다른 디렉토리를 선택하십시오.

   현재 존재하지 않는 디렉토리에 Message Service Client for .NET를 설치하도록 선택하면 설치 마법사가 디렉토리를 작성합니다.

   XMS 애플리케이션을 개발하려면 **개발 도구 및 샘플** 기능을 선택합니다. 이 기능은 샘플 애플리케이션, .NET 애플리케이션을 컴파일하는 데 필요한 라이브러리와 기타 파일을 제공합니다. 이 기능을 선택하지 않는 경우 XMS 애플리케이션을 실행하는 데 필요한 파일만 설치됩니다.
7. 사용자 정의 설치 옵션을 사용 중인 경우, 15 페이지의 『6』 단계에 설명된 대로 필요한 옵션을 선택한 후 **다음**을 클릭하십시오.

   설치 마법사에 설치를 시작할 준비가 되었음을 나타내는 메시지가 표시됩니다.
8. **설치**를 눌러 설치를 시작하십시오.

   설치 마법사에 설치 진행률을 표시하는 막대가 표시됩니다. 진행 표시줄이 완료될 때까지 기다리십시오. 설치가 완료되면 마법사에 다음 메시지가 표시됩니다.

   ```
   The installation wizard has successfully installed IBM Message Service Client
   for .NET. Click Finish to exit the wizard.
   ```
9. **마침**을 클릭하여 설치 마법사를 닫으십시오.

## 결과

Message Service Client for .NET를 설치했으며, 사용할 준비가 되었습니다.

## 다음에 수행할 작업

XMS에서 제공하는 샘플 애플리케이션을 포함하여 XMS 애플리케이션을 실행하기 전에, 메시징 서버 환경을 설정해야 합니다. 자세한 내용은 10 페이지의 『메시징 서버 환경 설정』의 내용을 참조하십시오.

**관련 개념**

JNDI 검색 웹 서비스

XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

메시징 서버 환경 설정

이 장에서는 XMS 애플리케이션이 서버에 연결할 수 있도록 메시징 서버 환경을 설정하는 방법에 대해 설명합니다.

XMS 샘플 애플리케이션 사용

XMS와 함께 제공된 샘플 애플리케이션을 사용하여 설치 및 메시징 서버 설정을 확인하고 고유한 애플리케이션을 빌드하도록 도와줍니다. 샘플은 각 API의 공통 기능에 대해 간략하게 설명합니다.

**관련 태스크**

WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성

이 섹션은 WebSphere MQ 버전 7.0 이상을 사용하고 있다고 가정합니다. WebSphere MQ 큐 관리자에 연결되는 애플리케이션을 실행하기 전에 큐 관리자를 구성해야 합니다. 발행/구독 애플리케이션의 경우 큐에 있는 발행/구독 인터페이스를 사용하는 경우 일부 추가 구성이 필요합니다.

브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성

브로커에 대한 실시간 연결을 사용하는 애플리케이션을 실행하려면 해당 브로커를 구성해야 합니다.

WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성
WebSphere 서비스 통합 버스에 연결되는 애플리케이션을 실행하기 전에, 기본 메시징 제공자를 사용하는 JMS 애플리케이션을 실행하도록 서비스 통합 버스를 구성하는 방식과 동일한 방법으로 서비스 통합 버스를 구성해야 합니다.

**관련 참조**

WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건
XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

## WebSphere MQ에 연결하는 XMS 애플리케이션의 필수 조건

XMS 애플리케이션이 WebSphere MQ에 연결하는 경우 일부 필수 조건이 적용됩니다.

WebSphere MQ 큐 관리자에 연결하는 애플리케이션의 경우, XMS 애플리케이션을 실행하는 데 사용하는 시스템에 적절한 WebSphere MQ 클라이언트 라이브러리를 설치해야 합니다. 이 라이브러리는 로컬 큐 관리자가 있는 시스템에 사전 설치됩니다.

.NET용 XMS 클라이언트의 경우, WebSphere MQ 버전 7.0.1.0 이상과 함께 제공된 클라이언트 라이브러리를 사용하십시오. 이는 *.NET*용 *WebSphere MQ* 클래스입니다. 버전 7.0.1.0 이상인 경우, 이를 통해 WebSphere MQ 버전 7.0, WebSphere MQ 버전 6.0 및 WebSphere MQ 버전 5.3 큐 관리자로의 클라이언트 모드 연결 및 로컬 큐 관리자로의 바인딩 모드 연결을 사용할 수 있습니다.

Microsoft .NET Framework 버전 2.0 Redistributable Package를 XMS를 설치할 컴퓨터에 설치해야 합니다. 이 패키지를 사용할 수 없는 경우, XMS 설치에 실패합니다. 그러면 설치 프로시저를 중단하고 컴퓨터에 Microsoft .NET Framework 버전 2.0 Redistributable Package를 설치한 후 설치 프로시저를 다시 실행해야 합니다.

Microsoft 다운로드 사이트에서, Microsoft .NET Framework 버전 2.0 Redistributable Package(x86)용 dotnetfx.exe 및 Microsoft .NET Framework 버전 2.0 Redistributable Package(x64)용 NetFx64.exe를 찾아야 하며, 어느 것이든 적용 가능합니다.

**관련 개념**

메시징 서버 환경 설정
이 장에서는 XMS 애플리케이션이 서버에 연결할 수 있도록 메시징 서버 환경을 설정하는 방법에 대해 설명합니다.

**관련 태스크**

WebSphere MQ 큐 관리자에 연결되는 애플리케이션의 큐 관리자와 브로커 구성
이 섹션은 WebSphere MQ 버전 7.0 이상을 사용하고 있다고 가정합니다. WebSphere MQ 큐 관리자에 연결되는 애플리케이션을 실행하기 전에 큐 관리자를 구성해야 합니다. 발행/구독 애플리케이션의 경우 큐에 있는 발행/구독 인터페이스를 사용하는 경우 일부 추가 구성이 필요합니다.

브로커에 대한 실시간 연결을 사용하는 애플리케이션의 브로커 구성
브로커에 대한 실시간 연결을 사용하는 애플리케이션을 실행하려면 해당 브로커를 구성해야 합니다.

WebSphere 서비스 통합 버스에 연결되는 애플리케이션의 서비스 통합 버스 구성
WebSphere 서비스 통합 버스에 연결되는 애플리케이션을 실행하기 전에, 기본 메시징 제공자를 사용하는 JMS 애플리케이션을 실행하도록 서비스 통합 버스를 구성하는 방식과 동일한 방법으로 서비스 통합 버스를 구성해야 합니다.

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

# XMS 샘플 애플리케이션 사용

XMS와 함께 제공된 샘플 애플리케이션을 사용하여 설치 및 메시징 서버 설정을 확인하고 고유한 애플리케이션을 빌드하도록 도와줍니다. 샘플은 각 API의 공통 기능에 대해 간략하게 설명합니다.

**관련 개념**

JNDI 검색 웹 서비스

XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

메시징 서버 환경 설정
이 장에서는 XMS 애플리케이션이 서버에 연결할 수 있도록 메시징 서버 환경을 설정하는 방법에 대해 설명합니다.

**관련 태스크**

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

# 샘플 애플리케이션

샘플 애플리케이션은 각 API의 공통 기능에 대해 간략하게 설명합니다. 이를 사용하여 설치 및 메시징 서버 설정을 확인하고 자신만의 애플리케이션을 빌드하는 데 도움을 받을 수 있습니다.

자체 애플리케이션을 작성하는 데 도움이 필요하면 시작점으로서 샘플 애플리케이션을 사용할 수 있습니다. 각 애플리케이션에 대해 소스 버전과 컴파일 버전 모두가 제공됩니다. 샘플 소스 코드를 검토하고 애플리케이션의 필수 오브젝트(ConnectionFactory, Connection, Session, Destination 및 Producer 또는 Consumer 또는 모두)를 작성하고 애플리케이션의 작업 방법을 지정하는 데 필요한 특정 특성을 설정하는 핵심 단계를 식별하십시오. 추가 정보는 19 페이지의 『XMS 애플리케이션 작성』의 내용을 참조하십시오. 샘플은 XMS의 이후 릴리스에서 변경됩니다.

다음 표에는 XMS와 함께 제공되는 샘플 애플리케이션(각 API용)의 3개 세트가 표시됩니다.

| 샘플 이름 | 설명 |
|---|---|
| SampleConsumerCS | 큐에서 메시지를 가져오거나 토픽을 구독하는 메시지 이용자 애플리케이션. |
| SampleProducerCS | 큐 또는 토픽에서 메시지를 생성하는 메시지 생성자 애플리케이션. |
| SampleConfigCS | 파일 기반의 관리 대상 오브젝트 저장소를 작성하는 데 사용할 수 있는 구성 애플리케이션. 애플리케이션에는 특정 연결 설정의 연결 팩토리 및 목적지가 포함됩니다. 이 관리 대상 오브젝트 저장소는 샘플 이용자 및 생성자 애플리케이션 각각에서 사용할 수 있습니다. |

다양한 API에서 동일한 기능을 지원하는 샘플에는 구문 상 차이가 있습니다.

- 샘플 메시지 이용자 및 생성자 애플리케이션은 모두 다음 기능을 지원합니다.
  - WebSphere MQ, IBM Integration Bus(브로커에 실시간 연결 사용) 및 WebSphere 서비스 통합 버스에 연결
  - 초기 컨텍스트 인터페이스를 사용하여 관리 대상 오브젝트 저장소 검색
  - 큐(WebSphere MQ 및 WebSphere 서비스 통합 버스) 및 토픽(WebSphere MQ, 브로커에 실시간 연결 및 WebSphere 서비스 통합 버스)에 연결
  - 기본, 바이트, 맵, 오브젝트, 스트림 및 텍스트 메시지
- 샘플 메시지 이용자 애플리케이션은 동기 및 비동기 수신 모드와 SQL 선택자 명령문을 지원합니다.
- 샘플 메시지 생성자 애플리케이션은 지속 전달 모드와 비지속 전달 모드를 지원합니다.

## 작동 모드

샘플은 두 모드 중 하나에서 작동할 수 있습니다.

**단순 모드**
최소 사용자 입력으로 샘플을 실행할 수 있습니다.

**고급 모드**
샘플이 작동하는 방식을 좀 더 세밀하게 사용자 정의할 수 있습니다.

모든 샘플은 호환 가능하므로 여러 언어로 운영될 수 있습니다.

**관련 개념**

자체 애플리케이션 빌드

샘플 애플리케이션을 빌드하는 것처럼 자체 애플리케이션을 빌드할 수 있습니다.

**관련 태스크**

샘플 애플리케이션 실행

.NET 샘플 애플리케이션을 단순 또는 고급 모드로 대화식으로 실행하거나, 자동 생성 또는 사용자 정의한 응답 파일을 사용하여 비대화식으로 실행할 수 있습니다.

.NET 샘플 애플리케이션 빌드

샘플 .NET 애플리케이션을 빌드하는 경우 선택한 샘플의 실행 파일이 작성됩니다.

# 샘플 애플리케이션 실행

.NET 샘플 애플리케이션을 단순 또는 고급 모드로 대화식으로 실행하거나, 자동 생성 또는 사용자 정의한 응답 파일을 사용하여 비대화식으로 실행할 수 있습니다.

## 시작하기 전에

제공된 샘플 애플리케이션을 실행하기 전에 애플리케이션이 서버에 연결할 수 있도록 먼저 메시징 서버 환경을 설정해야 합니다. 10 페이지의 『메시징 서버 환경 설정』을 참조하십시오.

## 프로시저

.NET 샘플 애플리케이션을 실행하려면 다음 단계를 완료하십시오.

**팁:** 샘플 애플리케이션을 실행할 때, 다음 수행할 작업에 대해 도움을 받으려면 언제든지 ?를 입력하십시오.

1. 샘플 애플리케이션을 실행할 모드를 선택하십시오.

   `Advanced` 또는 `Simple`을 입력하십시오.

2. 질문에 응답하십시오.

   질문 끝에서 대괄호로 묶여 표시되는 기본값을 선택하려면 Enter를 누르십시오. 다른 값을 선택하려면 적절한 값을 입력하고 Enter를 누르십시오.

   다음은 질문의 예입니다.

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

   이 경우 기본값은 `wpm`(WebSphere 서비스 통합 버스에 연결)입니다.

## 결과

샘플 애플리케이션을 실행하면 응답 파일이 현재 작업 디렉토리에서 자동으로 생성됩니다. 응답 파일 이름은 *`connection_type-sample_type`*`.rsp` 형식입니다. 예: `wpm-producer.rsp`. 필요한 경우 다시 옵션을 입력할 필요가 없도록 생성된 응답 파일을 사용하여 동일한 옵션으로 샘플 애플리케이션을 다시 실행할 수 있습니다.

**관련 개념**

샘플 애플리케이션

샘플 애플리케이션은 각 API의 공통 기능에 대해 간략하게 설명합니다. 이를 사용하여 설치 및 메시징 서버 설정을 확인하고 자신만의 애플리케이션을 빌드하는 데 도움을 받을 수 있습니다.

**관련 태스크**

.NET 샘플 애플리케이션 빌드

샘플 .NET 애플리케이션을 빌드하는 경우 선택한 샘플의 실행 파일이 작성됩니다.

## .NET 샘플 애플리케이션 빌드

샘플 .NET 애플리케이션을 빌드하는 경우 선택한 샘플의 실행 파일이 작성됩니다.

### 시작하기 전에

적절한 컴파일러를 설치하십시오. 이 작업을 하려면 Visual Studio 2012가 설치되어 있고 이를 잘 알고 있어야 합니다.

### 프로시저

.NET 샘플 애플리케이션을 빌드하려면 다음 단계를 완료하십시오.

1. .NET 샘플과 함께 제공된 `Samples.sln` 솔루션 파일을 클릭하십시오.
2. 솔루션 탐색기 창에서 솔루션 `Samples`을 마우스의 오른쪽 단추로 클릭하고 **솔루션 빌드**를 선택하십시오.

### 결과

선택한 구성에 따라 샘플의 적절한 서브폴더(`bin/Debug` 또는 `bin/Release` 중 하나)에서 실행 가능 프로그램이 작성됩니다. 이 프로그램의 이름은 폴더와 동일하며 접미부는 `CS`입니다. 예를 들어 메시지 생성자 샘플 애플리케이션의 C# 버전을 빌드하는 경우 `SampleProducerCS.exe`가 `SampleProducer` 폴더에서 작성됩니다.

**관련 개념**

샘플 애플리케이션
샘플 애플리케이션은 각 API의 공통 기능에 대해 간략하게 설명합니다. 이를 사용하여 설치 및 메시징 서버 설정을 확인하고 자신만의 애플리케이션을 빌드하는 데 도움을 받을 수 있습니다.

40 페이지의 『자체 애플리케이션 빌드』
샘플 애플리케이션을 빌드하는 것처럼 자체 애플리케이션을 빌드할 수 있습니다.

**관련 태스크**

샘플 애플리케이션 실행
.NET 샘플 애플리케이션을 단순 또는 고급 모드로 대화식으로 실행하거나, 자동 생성 또는 사용자 정의한 응답 파일을 사용하여 비대화식으로 실행할 수 있습니다.

# XMS 애플리케이션 개발

이 장에서는 XMS 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

XMS 애플리케이션 작성에 대한 정보는 다음 주제를 참조하십시오.

## XMS 애플리케이션 작성

이 장에서는 XMS 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

이 장에서는 XMS 애플리케이션 쓰기에 대한 일반 개념을 설명합니다. .NET 애플리케이션 작성에 대한 정보를 보려면 41 페이지의 『XMS .NET 애플리케이션 작성』의 내용을 참조하십시오.

이 장에는 다음 절가 포함됩니다.

- 37 페이지의 『한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환』
- 39 페이지의 『반복기』
- 40 페이지의 『코드화 문자 세트 ID』
- 40 페이지의 『XMS 오류 및 예외 코드』
- 40 페이지의 『자체 애플리케이션 빌드』

**관련 개념**

XMS .NET 애플리케이션 작성
이 장에서는 XMS.NET 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

**관련 참조**

.NET 인터페이스
이 절에서는 .NET 클래스 인터페이스와 해당 특성 및 메소드에 대해 설명합니다.

## 스레드 모델

멀티스레드 애플리케이션이 XMS 오브젝트를 사용하는 방법에 적용되는 일반 규칙입니다.

- 다음 유형의 오브젝트만 다른 스레드에서 동시에 사용할 수 있습니다.
  - ConnectionFactory
  - 연결
  - ConnectionMetaData
  - 목적지
- Session 오브젝트는 한 번에 하나의 스레드에서만 사용할 수 있습니다.

이 규칙의 예외는 API 참조 장에 있는 메소드의 인터페이스 정의에서 "스레드 컨텍스트"라는 항목으로 표시됩니다.

## 런타임 시 핸들링할 수 있는 오류 조건

API 호출의 리턴 코드는 런타임 시 핸들링할 수 있는 오류 조건입니다. 이 유형의 오류를 처리하는 방식은 C 또는 C++ API를 사용하는지에 따라 다릅니다.

## 런타임 시 오류를 발견하는 방법

애플리케이션이 C API 함수를 호출하고 호출에 실패한 경우, 실패 이유에 대한 자세한 정보가 들어 있는 XMS 오류 블록과 함께 XMS_OK가 아닌 리턴 코드가 포함된 응답이 리턴됩니다.

C++ API는 메소드가 사용될 때 예외를 처리합니다.

애플리케이션은 예외 리스너를 사용하여 연결 문제점에 대해 비동기로 알림을 받습니다. 예외 리스너에게 제공되고 XMS C 또는 C++ API를 사용하여 예외 리스너가 초기화됩니다.

## 런타임 시 오류를 핸들링하는 방법

일부 오류 조건은 일부 자원이 사용 불가능함을 나타내는 표시이며 애플리케이션이 수행할 수 있는 조치는 애플리케이션이 호출하는 XMS 함수에 따라 다릅니다. 예를 들어, 서버에 연결하는 데 실패하면 애플리케이션은 연결이 될 때까지 주기적으로 재시도할 수 있습니다. XMS 오류 블록 또는 예외에 수행 조치를 결정할 수 있는 정보가 충분히 포함되지 않을 수 있으며 이런 경우 특정 진단 정보가 포함된 링크 오류 블록 또는 예외가 있을 수 있습니다.

C API에서는 항상 XMS_OK 이외의 리턴 코드가 있는 응답을 테스트하고 API 호출 시 오류 블록을 전달하십시오. 수행된 조치는 애플리케이션에서 사용하는 API 함수에 따라 다릅니다.

C++ API에서는 항상 try 블록에 메소드 호출을 포함시키십시오. 모든 유형의 XMS 예외를 발견하려면 발견 구성에 예외(Exception) 클래스를 지정하십시오.

예외 리스너는 언제든지 시작할 수 있는 비동기 오류 조건 경로입니다. 예외 리스너 함수가 자체의 스레드에서 시작되면 일반적인 XMS API 오류 조건보다 심각한 장애가 있다는 표시입니다. 적절한 조치가 수행될 수 있지만 20 페이지의 『스레드 모델』에 설명된 바와 같이 XMS 스레드 모델에 대한 규칙을 따라야 합니다.

## ConnectionFactories 오브젝트 및 Connection 오브젝트

ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

.NET의 경우 XMS 애플리케이션은 먼저 XMSFactoryFactory 오브젝트를 사용하여 필요한 프로토콜 유형에 적절한 ConnectionFactory 오브젝트에 대한 참조를 가져옵니다. 이 ConnectionFactory 오브젝트는 해당 프로토콜 유형에 대해서만 연결을 생성할 수 있습니다.

XMS 애플리케이션은 다중 연결을 작성할 수 있고 멀티스레드 애플리케이션은 단일 Connection 오브젝트를 다중 스레드에서 동시에 사용할 수 있습니다. Connection 오브젝트는 애플리케이션과 메시징 서버 사이의 통신 연결을 캡슐화합니다.

연결은 다음과 같은 여러 가지 용도로 사용됩니다.

- 애플리케이션에서 연결을 작성하면 애플리케이션은 인증 받을 수 있습니다.
- 애플리케이션은 고유한 클라이언트 ID를 연결과 연관시킬 수 있습니다. 클라이언트 ID를 사용하여 발행/구독 도메인에서 지속 가능한 구독을 지원할 수 있습니다. 클라이언트 ID는 두 가지 방법으로 설정할 수 있습니다.

  연결 클라이언트 ID를 지정하는 바람직한 방법은 특성을 사용하여 클라이언트 고유의 ConnectionFactory 오브젝트에서 구성하고 이를 투명하게 작성되는 연결에 지정하는 것입니다.

  다른 방법으로는, Connection 오브젝트에 설정되어 있는 제공자 고유 값을 사용하여 클라이언트 ID를 지정하는 방법이 있습니다. 이 값은 관리상 구성된 ID를 대체하지 않습니다. 관리상 지정된 ID가 존재하지 않는 경우에 제공됩니다. 관리상 지정된 ID가 존재하는 경우 제공자 고유 값으로 대체하려 시도하면 예외가 발생합니다. 애플리케이션이 ID를 명시적으로 설정하는 경우 이는 연결을 작성한 후 그리고 연결에 대한 다른 조치를 수행하기 바로 전에 수행해야 합니다. 그렇지 않으면 예외가 발생합니다.

XMS 애플리케이션은 일반적으로 한 개의 연결, 한 개 이상의 세션, 메시지 생성자 및 메시지 이용자 수를 작성합니다.

연결 작성은 통신 연결을 생성하고 애플리케이션을 인증하는 작업과도 관련될 수 있으므로 시스템 자원 측면에서는 상당히 소모적입니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 연결 시작됨 및 중지됨 모드

연결은 시작됨 또는 중지됨 모드에서 작동될 수 있습니다.

애플리케이션에서 연결을 작성하면 연결은 중지됨 모드가 됩니다. 연결이 중지됨 모드인 경우 애플리케이션은 세션을 초기화할 수 있고 동기적 또는 비동기적으로 메시지를 전송할 수 있으나 수신할 수 없습니다.

애플리케이션은 `Start Connection` 메소드를 호출하여 연결을 시작할 수 있습니다. 연결이 시작됨 모드인 경우 애플리케이션은 메시지를 전송하거나 수신할 수 있습니다. 그런 다음 애플리케이션은 Stop Connection 및 `Start Connection` 메소드를 호출하여 연결을 중지한 후 다시 시작할 수 있습니다.

**관련 개념**

연결 닫기
애플리케이션은 Close Connection 메소드를 호출하여 연결을 닫습니다.

예외 처리
애플리케이션이 비동기로 메시지를 이용하는 용도로만 연결을 사용하는 경우 연결에 대한 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

## 연결 닫기

애플리케이션은 Close Connection 메소드를 호출하여 연결을 닫습니다.

애플리케이션에서 연결을 닫는 경우 XMS는 다음 조치를 수행합니다.

- 연결과 연관된 세션을 모두 닫고 해당 세션과 연관된 특정 오브젝트를 삭제합니다. 삭제되는 오브젝트에 대한 자세한 정보는 36 페이지의 『오브젝트 삭제』의 내용을 참조하십시오. 동시에 XMS는 세션 내에서 현재 진행 중인 모든 트랜잭션을 롤백합니다.
- 메시징 서버와의 통신 연결을 종료합니다.
- 연결에서 사용된 메모리 및 기타 내부 자원을 해제합니다.

XMS는 연결을 닫기 전, 세션 중에 수신확인에 실패한 메시지의 수신을 수신확인하지 않습니다. 메시지 수신 수신확인에 대한 자세한 정보는 24 페이지의 『메시지 수신확인』의 내용을 참조하십시오.

**관련 개념**
연결 시작됨 및 중지됨 모드
연결은 시작됨 또는 중지됨 모드에서 작동될 수 있습니다.

예외 처리
애플리케이션이 비동기로 메시지를 이용하는 용도로만 연결을 사용하는 경우 연결에 대한 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

## 예외 처리

애플리케이션이 비동기로 메시지를 이용하는 용도로만 연결을 사용하는 경우 연결에 대한 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다.

XMS .NET 예외는 모두 System.Exception에서 파생됩니다. 자세한 정보는 44 페이지의 『.NET의 오류 처리』의 내용을 참조하십시오.

**관련 개념**
연결 시작됨 및 중지됨 모드
연결은 시작됨 또는 중지됨 모드에서 작동될 수 있습니다.

연결 닫기
애플리케이션은 Close Connection 메소드를 호출하여 연결을 닫습니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

## *WebSphere* 서비스 통합 버스에 대한 연결

직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

TCP/IP와의 직접 연결이 불가능한 경우 HTTP 프로토콜을 사용할 수 있습니다. 일반적인 상황은 예를 들어, 두 기업이 메시지를 교환하는 경우와 같이 방화벽을 통해 통신하는 경우입니다. HTTP를 사용하여 방화벽을 통해 통신하는 것을 *HTTP* 터널링이라고 합니다. 그러나 HTTP 터널링은 직접 TCP/IP 연결을 사용하는 것보다 본질적으로 느립니다. HTTP 헤더는 전송되는 데이터 양에 상당하게 추가되고 HTTP 프로토콜에서는 TCP/IP보다 더 많은 통신 플로우가 필요하기 때문입니다.

TCP/IP 연결을 작성하기 위해 애플리케이션은 XMSC_WPM_TARGET_TRANSPORT_CHAIN 특성이 XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC으로 설정된 연결 팩토리를 사용할 수 있습니다. 이것이 특성의 기본값입니다. 연결이 작성되면 연결의 XMSC_WPM_CONNECTION_PROTOCOL 특성이 XMSC_WPM_CP_TCP로 설정됩니다.

HTTP를 사용하는 연결을 작성하려면, 애플리케이션이 HTTP 전송 채널을 사용하도록 구성된 인바운드 전송 체인의 이름으로 XMSC_WPM_TARGET_TRANSPORT_CHAIN 특성이 설정된 연결 팩토리를 사용해야 합니다. 연결이 작성되면 연결의 XMSC_WPM_CONNECTION_PROTOCOL 특성이 XMSC_WPM_CP_TCP로 설정됩니다. 전송 체인 구성 방법에 대한 정보는 WebSphere(r) Application Server 제품 문서의 전송 체인 구성을 참조하십시오.

부트스트랩 서버에 연결할 경우에도 애플리케이션은 유사한 통신 프로토콜을 선택하게 됩니다. 연결 팩토리의 XMSC_WPM_PROVIDER_ENDPOINTS 특성은 하나 이상의 부트스트랩 서버 엔드포인트 주소입니다. 각 엔드포인트 주소의 부트스트랩 전송 체인 컴포넌트는 부트스트랩 서버와 TCP/IP로 연결될 경우에는 XMSC_WPM_BOOTSTRAP_TCP이며, HTTP를 사용하는 연결일 경우에는 XMSC_WPM_BOOTSTRAP_HTTP입니다.

**관련 개념**

연결 시작됨 및 중지됨 모드
연결은 시작됨 또는 중지됨 모드에서 작동될 수 있습니다.

연결 닫기
애플리케이션은 Close Connection 메소드를 호출하여 연결을 닫습니다.

예외 처리
애플리케이션이 비동기로 메시지를 이용하는 용도로만 연결을 사용하는 경우 연결에 대한 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 세션

세션은 메시지 송신 및 수신을 위한 단일 스레드 컨텍스트입니다.

애플리케이션은 세션을 사용하여 메시지, 메시지 생성자, 메시지 이용자, 큐 브라우저 및 임시 목적지를 작성할 수 있습니다. 또한 세션을 사용하여 로컬 트랜잭션을 실행할 수도 있습니다.

애플리케이션은 다중 세션을 작성할 수 있으며 각 세션은 다른 세션과 독립적으로 메시지를 생성하고 이용합니다. 별도의 세션(또는 동일한 세션)에 있는 두 메시지 이용자가 같은 토픽을 구독하면 각각 해당 토픽에 발행된 임의의 메시지의 사본을 수신합니다.

Connection 오브젝트와 달리 Session 오브젝트는 다른 스레드에서 동시에 사용할 수 없습니다. Session 오브젝트의 Close Session 메소드만 당시 Session 오브젝트가 사용 중인 스레드 이외의 스레드에서 호출할 수 있습니다. Close Session 메소드는 세션을 끝내고 세션에 할당된 시스템 자원을 릴리스합니다.

애플리케이션이 둘 이상의 스레드에서 동시에 메시지를 처리해야 하는 경우, 애플리케이션은 각 스레드에서 하나의 세션을 작성하고, 해당 스레드 내에서의 전송 또는 수신 조작에 해당 세션을 사용해야 합니다.

## 트랜잭션 세션

XMS 애플리케이션은 로컬 트랜잭션을 실행할 수 있습니다. 로컬 트랜잭션은 애플리케이션이 연결된 큐 관리자나 서비스 통합 버스의 자원에 대한 변경사항만 포함하는 트랜잭션입니다.

이 절의 정보는 애플리케이션이 WebSphere MQ 큐 관리자 또는 WebSphere 서비스 통합 버스에 연결하는 경우에만 관련됩니다. 정보는 브로커에 대한 실시간 연결과 관련이 없습니다.

로컬 트랜잭션을 실행하기 위해 애플리케이션은 Connection 오브젝트의 Create Session 메소드를 호출하고 세션이 트랜잭션되는 매개변수를 지정하여 트랜잭트되는 세션을 먼저 작성해야 합니다. 결과적으로, 세션 내에서 송신 및 수신된 모든 메시지는 트랜잭션의 시퀀스로 그룹화됩니다. 트랜잭션이 시작된 이후에 송신하거나 수신한 메시지를 애플리케이션이 커미트하거나 롤백하면 트랜잭션이 종료됩니다.

트랜잭션을 커미트하기 위해 애플리케이션은 Session 오브젝트의 Commit 메소드를 호출합니다. 트랜잭션이 커미트되는 경우, 트랜잭션 내에서 송신된 모든 메시지가 다른 애플리케이션에 전달하기 위해 사용 가능하며 트랜잭션 내에서 수신된 모든 메시지가 수신확인되므로 메시징 서버는 이를 다시 애플리케이션에 전달하려고 시도하지 않습니다. 포인트-투-포인트 도메인에서, 메시징 서버는 해당 큐에서 수신된 메시지도 제거합니다.

트랜잭션을 롤백하기 위해 애플리케이션은 Session 오브젝트의 Rollback 메소드를 호출합니다. 트랜잭션이 롤백되면 메시징 서버는 트랜잭션 내의 송신된 모든 메시지를 버리며, 트랜잭션 내의 수신된 모든 메시지가 다시 전달을 위해 사용 가능합니다. 포인트-투-포인트 도메인에서, 수신된 메시지는 해당 큐에 다시 놓이며 다시 기타 애플리케이션에서 이를 볼 수 있습니다.

애플리케이션이 트랜잭트된 세션을 작성하거나 Commit 또는 Rollback 메소드를 호출하면 새로운 트랜잭션이 자동으로 시작됩니다. 따라서 트랜잭션 세션에는 항상 활성 트랜잭션이 있습니다.

애플리케이션이 트랜잭션 세션을 닫으면 암시적 롤백이 발생합니다. 애플리케이션이 연결을 닫으면 모든 연결의 트랜잭트 세션에 대해 암시적 롤백이 발생합니다.

트랜잭션은 전적으로 트랜잭션 세션 내에 포함됩니다. 트랜잭션은 세션 간에 걸칠 수 없습니다. 이는 애플리케이션이 두 개 이상의 트랜잭션 세션에서 메시지를 송신하고 수신한 후에 이 모든 조치를 단일 트랜잭션으로 커미트하거나 롤백할 수 없음을 의미합니다.

**관련 개념**

메시지 수신확인
트랜잭션되지 않은 모든 세션에는 애플리케이션이 수신한 메시지가 수신확인되는 방법을 결정하는 수신확인 모드가 있습니다. 3개의 수신확인 모드가 사용 가능하며, 수신확인 모드의 선택사항은 애플리케이션의 디자인에 영향을 줍니다.

비동기 메시지 전달
XMS에서는 하나의 스레드를 사용하여 세션에 대한 모든 비동기 메시지 전달을 처리합니다. 이는 한 번에 하나의 메시지 리스너 기능 또는 하나의 onMessage() 메소드만 실행할 수 있음을 의미합니다.

동기 메시지 전달
애플리케이션이 MessageConsumer 오브젝트의 Receive 메소드를 사용할 경우 메시지는 애플리케이션에 동기로 전달됩니다.

메시지 전달 모드
XMS는 두 가지 메시지 전달 모드를 지원합니다.

## 메시지 수신확인

트랜잭션되지 않은 모든 세션에는 애플리케이션이 수신한 메시지가 수신확인되는 방법을 결정하는 수신확인 모드가 있습니다. 3개의 수신확인 모드가 사용 가능하며, 수신확인 모드의 선택사항은 애플리케이션의 디자인에 영향을 줍니다.

이 절의 정보는 애플리케이션이 WebSphere MQ 큐 관리자 또는 WebSphere 서비스 통합 버스에 연결하는 경우에만 관련됩니다. 정보는 브로커에 대한 실시간 연결과 관련이 없습니다.

XMS는 JMS가 사용하는 메시지의 수신 수신확인에 동일한 메커니즘을 사용합니다.

세션이 트랜잭션되지 않은 경우, 애플리케이션이 수신한 메시지가 수신확인되는 방법은 세션의 수신확인 모드에 의해 판별됩니다. 다음 단락에서는 세 개의 수신확인 모드가 설명되어 있습니다.

**XMSC_AUTO_ACKNOWLEDGE**
세션은 애플리케이션이 수신한 각 메시지를 자동으로 수신확인합니다.

메시지가 애플리케이션에 동기적으로 전달되는 경우, 세션은 수신 호출이 완료될 때마다 메시지의 수신을 수신확인합니다.

애플리케이션이 메시지를 성공적으로 수신하지만 장애로 인해 수신확인이 발생하지 않는 경우, 해당 메시지는 다시 전달에 사용될 수 있습니다. 따라서 애플리케이션은 다시 전달되는 메시지를 처리할 수 있어야 합니다.

**XMSC_DUPS_OK_ACKNOWLEDGE**
세션은 선택된 시간에 애플리케이션이 수신한 메시지를 수신확인합니다.

이 수신확인 모드를 사용하면 세션이 수행해야 하는 작업의 양이 줄어들지만, 메시지 수신확인을 방지하는 장애의 결과로 인해 둘 이상의 메시지가 다시 전달에 사용될 수 있습니다. 따라서 애플리케이션은 다시 전달되는 메시지를 처리할 수 있어야 합니다.

**XMSC_CLIENT_ACKNOWLEDGE**
애플리케이션은 Message 클래스의 Acknowledge 메소드를 호출하여 수신한 메시지를 수신확인합니다.

애플리케이션은 각 메시지의 수신을 개별적으로 수신확인할 수도 있으며, 또는 메시지의 일괄처리를 수신하고 수신한 마지막 메시지에 대해서만 Acknowledge 메소드를 호출할 수도 있습니다. Acknowledge 메소드가 호출되는 경우, 메소드가 마지막으로 호출된 후에 수신된 모든 메시지가 수신확인됩니다.

이러한 수신확인 모드와 결합하여, 애플리케이션은 Session 클래스의 Recover 메소드를 호출하여 세션에서 메시지의 전달을 중지하고 다시 시작할 수 있습니다. 수신자가 이전에 수신확인하지 않은 메시지가 다시 전달됩니다. 그러나 이는 이전에 전달되었던 동일한 순서대로 전달되지 않을 수 있습니다. 그 동안에 보다 높은 우선순위의 메시지가 도착할 수도 있으며, 원래 메시지 중 일부가 만료될 수도 있습니다. 또한 포인트-투-포인트 도메인에서 원래 메시지의 일부를 다른 애플리케이션이 이용했을 수도 있습니다.

애플리케이션은 메시지의 JMSRedelivered 헤더 필드의 컨텐츠를 검사하여 메시지를 재전달하는지 여부를 판별할 수 있습니다. 애플리케이션은 Message 클래스의 Get JMSRedelivered 메소드를 호출하여 메시지의 재전달 여부를 결정합니다.

**관련 개념**
트랜잭션 세션
XMS 애플리케이션은 로컬 트랜잭션을 실행할 수 있습니다. 로컬 트랜잭션은 애플리케이션이 연결된 큐 관리자나 서비스 통합 버스의 자원에 대한 변경사항만 포함하는 트랜잭션입니다.

비동기 메시지 전달
XMS에서는 하나의 스레드를 사용하여 세션에 대한 모든 비동기 메시지 전달을 처리합니다. 이는 한 번에 하나의 메시지 리스너 기능 또는 하나의 onMessage() 메소드만 실행할 수 있음을 의미합니다.

동기 메시지 전달
애플리케이션이 MessageConsumer 오브젝트의 Receive 메소드를 사용할 경우 메시지는 애플리케이션에 동기로 전달됩니다.

메시지 전달 모드
XMS는 두 가지 메시지 전달 모드를 지원합니다.

## 비동기 메시지 전달
XMS에서는 하나의 스레드를 사용하여 세션에 대한 모든 비동기 메시지 전달을 처리합니다. 이는 한 번에 하나의 메시지 리스너 기능 또는 하나의 onMessage() 메소드만 실행할 수 있음을 의미합니다.

세션에 있는 둘 이상의 메시지 이용자가 메시지를 비동기식으로 수신하고 메시지 리스너 기능 또는 onMessage() 메소드가 메시지를 메시지 이용자에게 전달하는 경우, 동일한 메시지를 대기 중인 기타 메시지 이용자는 계속 대기해야 합니다. 세션으로 전달되기를 대기하는 다른 메시지도 계속 대기해야 합니다.

애플리케이션에서 메시지를 동시에 전달해야 하는 경우, XMS에서 둘 이상의 스레드를 사용하여 비동기식으로 메시지 전달을 처리할 수 있도록 둘 이상의 세션을 작성하십시오. 이러한 방식으로 하나 이상의 메시지 리스너 기능 또는 onMessage() 메소드는 동시에 실행될 수 있습니다.

이용자에게 메시지 리스너를 지정하면 세션이 비동기가 되지 않습니다. Connection.Start 메소드가 호출되는 경우에만 세션이 비동기가 됩니다. Connection.Start 메소드가 호출될 때까지 모든 동기 호출이 허용됩니다. Connection.Start 가 호출될 때 이용자에게 메시지 전달이 시작됩니다.

동기 호출 (예: 이용자 또는 생성자 작성) 이 비동기 세션에서 작성되어야 하는 경우 `Connection.Stop` 를 호출해야 합니다. `Connection.Start` 메소드를 호출하여 메시지 전달을 시작하여 세션을 재개할 수 있습니다. 이에 대한 유일한 예외는 콜백 기능으로 메시지를 전달하는 스레드인 세션 메시지 전달 스레드입니다. 이 스레드는 메시지 콜백 기능에서 세션에 대한 호출(닫기 호출 제외)을 작성할 수 있습니다.

**참고:** 비관리 모드에서, 콜백 함수의 MQDISC 호출은 WMQ.NET 클라이언트에서 지원되지 않습니다. 그러므로 클라이언트 애플리케이션은 비동기 수신 모드의 MessageListener 콜백 내에서 세션을 작성하거나 닫을 수 없습니다. `MessageListener` 메소드 외부에서 세션을 작성하고 처리하십시오.

**관련 개념**

트랜잭션 세션

XMS 애플리케이션은 로컬 트랜잭션을 실행할 수 있습니다. 로컬 트랜잭션은 애플리케이션이 연결된 큐 관리자나 서비스 통합 버스의 자원에 대한 변경사항만 포함하는 트랜잭션입니다.

메시지 수신확인

트랜잭션되지 않은 모든 세션에는 애플리케이션이 수신한 메시지가 수신확인되는 방법을 결정하는 수신확인 모드가 있습니다. 3개의 수신확인 모드가 사용 가능하며, 수신확인 모드의 선택사항은 애플리케이션의 디자인에 영향을 줍니다.

동기 메시지 전달

애플리케이션이 MessageConsumer 오브젝트의 Receive 메소드를 사용할 경우 메시지는 애플리케이션에 동기로 전달됩니다.

메시지 전달 모드

XMS는 두 가지 메시지 전달 모드를 지원합니다.

## 동기 메시지 전달

애플리케이션이 MessageConsumer 오브젝트의 Receive 메소드를 사용할 경우 메시지는 애플리케이션에 동기로 전달됩니다.

애플리케이션은 Receive 메소드를 사용하여 지정된 기간 동안 메시지를 대기하거나 무기한 대기할 수 있습니다. 또는 애플리케이션이 메시지를 대기하지 않을 경우 Receive with No Wait 메소드를 사용할 수 있습니다.

**관련 개념**

트랜잭션 세션

XMS 애플리케이션은 로컬 트랜잭션을 실행할 수 있습니다. 로컬 트랜잭션은 애플리케이션이 연결된 큐 관리자나 서비스 통합 버스의 자원에 대한 변경사항만 포함하는 트랜잭션입니다.

메시지 수신확인

트랜잭션되지 않은 모든 세션에는 애플리케이션이 수신한 메시지가 수신확인되는 방법을 결정하는 수신확인 모드가 있습니다. 3개의 수신확인 모드가 사용 가능하며, 수신확인 모드의 선택사항은 애플리케이션의 디자인에 영향을 줍니다.

비동기 메시지 전달

XMS에서는 하나의 스레드를 사용하여 세션에 대한 모든 비동기 메시지 전달을 처리합니다. 이는 한 번에 하나의 메시지 리스너 기능 또는 하나의 `onMessage()` 메소드만 실행할 수 있음을 의미합니다.

메시지 전달 모드

XMS는 두 가지 메시지 전달 모드를 지원합니다.

## 메시지 전달 모드

XMS는 두 가지 메시지 전달 모드를 지원합니다.

- 지속적 메시지는 한 번 전달됩니다. 메시징 서버는 메시지 로깅과 같은 특수 예방 조치를 수행하여 실패한 경우에도 전송 중인 지속적 메시지가 손실되지 않도록 합니다.
- 비지속적 메시지는 두 번 이상 전달되지 않습니다. 비지속 메시지는 실패 시 전송 중에 손실될 수 있으므로 지속 메시지보다 덜 안정적입니다.

전달 모드를 선택하면 신뢰도와 성능이 상충됩니다. 일반적으로 비지속 메시지는 지속 메시지보다 빨리 전송됩니다.

**관련 개념**

트랜잭션 세션

XMS 애플리케이션은 로컬 트랜잭션을 실행할 수 있습니다. 로컬 트랜잭션은 애플리케이션이 연결된 큐 관리자나 서비스 통합 버스의 자원에 대한 변경사항만 포함하는 트랜잭션입니다.

메시지 수신확인
트랜잭션되지 않은 모든 세션에는 애플리케이션이 수신한 메시지가 수신확인되는 방법을 결정하는 수신확인 모드가 있습니다. 3개의 수신확인 모드가 사용 가능하며, 수신확인 모드의 선택사항은 애플리케이션의 디자인에 영향을 줍니다.

비동기 메시지 전달
XMS에서는 하나의 스레드를 사용하여 세션에 대한 모든 비동기 메시지 전달을 처리합니다. 이는 한 번에 하나의 메시지 리스너 기능 또는 하나의 `onMessage()` 메소드만 실행할 수 있음을 의미합니다.

동기 메시지 전달
애플리케이션이 MessageConsumer 오브젝트의 Receive 메소드를 사용할 경우 메시지는 애플리케이션에 동기로 전달됩니다.

## 목적지

XMS 애플리케이션은 Destination 오브젝트를 사용하여 전송 중인 메시지의 목적지와 수신 중인 메시지의 소스를 지정합니다.

XMS 애플리케이션은 런타임 시 Destination 오브젝트를 작성하거나 관리 대상 오브젝트 저장소에서 사전 정의된 목적지를 가져올 수 있습니다.

ConnectionFactory의 경우와 마찬가지로 XMS 애플리케이션이 목적지를 지정하는 가장 유연한 방법은 목적지를 관리 대상 오브젝트로 정의하는 것입니다. 이 방법을 사용하여 C, C++, .NET 언어 및 Java로 작성된 애플리케이션이 대상 정의를 공유할 수 있습니다. 코드를 변경하지 않고도 관리 대상 Destination 오브젝트의 특성을 변경할 수 있습니다.

.NET애플리케이션의 경우 CreateTopic 또는 CreateQueue 메소드를 사용하여 목적지를 작성합니다. 이러한 두 메소드는 .NET API의 ISession 및 XMSFactoryFactory 오브젝트 모두에서 사용 가능합니다. 자세한 정보는 43 페이지의 『.NET의 목적지』 및 99 페이지의 『IDestination』의 내용을 참조하십시오.

**관련 참조**

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

### 토픽 *URI(Uniform Resource Identifier)*

토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

토픽 URI는 순서 토픽://로 시작하며 그 뒤에 토픽 이름이 옵니다. 선택적으로 나머지 토픽 특성을 설정하는 이름-값 쌍 목록이 뒤에 올 수 있습니다. 토픽 이름은 공백이 될 수 없습니다.

다음은 .NET 코드의 단편으로 된 예제입니다.

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

URI에서 사용할 수 있는 이름과 올바른 값을 포함하여 토픽의 특성에 대한 자세한 정보는 174 페이지의 『Destination 특성』의 내용을 참조하십시오.

구독에 사용할 토픽 URI를 지정할 때 와일드카드를 사용할 수 있습니다. 이러한 와일드카드의 구문은 연결 유형과 브로커 버전에 따라 달라집니다. 사용할 수 있는 옵션은 다음과 같습니다.

- 문자 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자
- 토픽 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자
- WebSphere 서비스 통합 버스

### 문자 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자

문자 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자는 다음과 같은 와일드카드 문자를 사용합니다.

* 0 이상의 문자
? 1개의 문자
% 이스케이프 문자

28 페이지의 표 1에는 이 와일드카드 설계의 사용 방법에 대한 몇 가지 예가 있습니다.

표 *1.* 예*: WebSphere MQ V7.0* 큐 관리자에 문자 레벨 와일드카드 설계를 사용하는 *URI*

| **URI(Uniform Resource Identifier)** | **일치사항** | **예:** |
|---|---|---|
| "topic://Sport*Results" | "Sport"로 시작하고 "Results"로 끝나는 모든 토픽 | "topic://SportsResults" and "topic://Sport/Hockey/National/Div3/Results" |
| "topic://Sport?Results" | "Sport"로 시작하고 그 뒤에 단일 문자와 "Results"가 차례로 오는 모든 토픽 | "topic://SportsResults" and "topic://SportXResults" |
| "topic://Sport/*ball*/Div?/Results/*/???" | 토픽 | "topic://Sport/Football/Div1/Results/2002/Nov" 및 "topic://Sport/Netball/National/Div3/Results/02/Jan" |

### 토픽 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자

토픽 레벨 와일드카드 형식의 WebSphere MQ V7.0 큐 관리자는 다음과 같은 와일드카드 문자를 사용합니다.

# 다중 레벨 일치
+ 단일 레벨 일치

28 페이지의 표 2에는 이 와일드카드 설계의 사용 방법에 대한 몇 가지 예가 있습니다.

표 *2.* 예*: WebSphere MQ V7.0* 큐 관리자에 토픽 레벨 와일드카드 설계를 사용하는 *URI*

| **URI(Uniform Resource Identifier)** | **일치사항** | **예:** |
|---|---|---|
| "topic://Sport/+/Results" | Sport와 Results 사이에 단일 계층 구조 레벨 이름이 있는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Ju-Jitsu/Results" |
| "topic://Sport/#/Results" | "Sport/"로 시작하고 "/Results"로 끝나는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Hockey/National/Div3/Results" |
| "topic://Sport/Football/#" | "Sport/Football/"로 시작하는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Football/TeamNews/Signings/Managerial" |

### WebSphere 서비스 통합 버스

WebSphere 서비스 통합 버스는 다음과 같은 와일드카드 문자를 사용합니다.

* 계층 구조에서 한 레벨의 임의의 문자 일치
// 0 이상의 레벨 일치
//. 0 이상의 레벨 일치(토픽 표현식 끝에서)

29 페이지의 표 3에는 이 와일드카드 설계의 사용 방법에 대한 몇 가지 예가 있습니다.

| 표 3. 예: *WebSphere* 서비스 통합 버스에 와일드카드 설계를 사용하는 *URI* | | |
|---|---|---|
| **URI(Uniform Resource Identifier)** | **일치사항** | **예:** |
| "topic://Sport/*ball/Results" | Sport와 Results 사이에 이름이 "ball"로 끝나는 단일 계층 구조 레벨이 있는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Netball/Results" |
| "topic://Sport//Results" | "Sport/"로 시작하고 "/Results"로 끝나는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Hockey/National/Div3/Results" |
| "topic://Sport/Football//." | "Sport/Football/"로 시작하는 모든 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Football/TeamNews/Signings/Managerial" |
| "topic://Sport/*ball//Results//." | 토픽 | "topic://Sport/Football/Results" 및 "topic://Sport/Netball/National/Div3/Results/2002/November" |

**관련 개념**

큐 URI(Uniform Resource Identifier)
큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

임시 목적지
XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

목적지 와일드 카드
XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

**관련 참조**

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 큐 *URI(Uniform Resource Identifier)*

큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

큐 URI는 시퀀스 큐://로 시작하며 그 뒤에 큐 이름이 옵니다. 또한, 나머지 큐 특성을 설정하는 이름-값 쌍 목록이 포함될 수 있습니다.

WebSphere MQ 큐의 경우(WebSphere 애플리케이션 서버 기본 메시징 제공자 큐의 경우는 아님), 큐가 상주하는 큐 관리자를 큐 앞에 지정할 수 있으며, 이 경우 /로 큐 관리자 이름과 큐 이름을 구분합니다.

큐 관리자가 지정되면 큐 관리자는 XMS가 이 큐를 사용하여 직접 연결되는 큐 관리자이거나 이 큐에서 액세스할 수 있어야 합니다. 리모트 큐 관리자는 큐에서 메시지를 검색하는 경우에만 지원되고 메시지를 큐에 넣는 경우에는 지원되지 않습니다. 전체 내용은 WebSphere MQ 큐 관리자 문서를 참조하십시오.

큐 관리자가 지정되지 않는 경우 추가 / 구분 기호는 선택사항이며 구분 기호가 있거나 없어도 큐 정의에는 차이가 없습니다.

다음 큐 정의는 모두 XMS가 직접 연결되는 큐 관리자(QM_A)의 WebSphere MQ 큐(QB)에 대해 같습니다.

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

**관련 개념**

토픽 URI(Uniform Resource Identifier)
토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

임시 목적지
XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

목적지 와일드 카드
XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

**관련 참조**

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 임시 목적지

XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

애플리케이션은 일반적으로 임시 목적지를 사용하여 요청 메시지에 대한 응답을 수신합니다. 요청 메시지에 대한 응답을 보낼 목적지를 지정하기 위해 애플리케이션은 요청 메시지를 표시하는 Message 오브젝트의 Set JMSReplyTo 메소드를 호출합니다. 호출에 지정한 목적지는 임시 목적지가 될 수 있습니다.

임시 목적지를 작성하는 데 세션이 사용되지만 실제로 임시 목적지의 범위는 세션을 작성하는 데 사용된 연결입니다. 연결의 세션은 임시 목적지에 대한 메시지 생성자와 메시지 이용자를 작성할 수 있습니다. 임시 목적지는 명시적으로 삭제되거나 연결이 종료되거나 먼저 발생하는 때까지 그대로 유지됩니다.

애플리케이션이 임시 큐를 작성할 때 애플리케이션이 연결되어 있는 메시징 서버에서 큐가 작성됩니다. 애플리케이션이 큐 관리자에 연결되는 경우 XMSC_WMQ_TEMPORARY_MODEL 특성에서 이름이 지정되는 모델 큐에서 동적 큐가 작성되고 동적 큐의 이름을 구성하는 데 사용되는 접두부는 XMSC_WMQ_TEMP_Q_PREFIX 특성에서 지정됩니다. 애플리케이션이 서비스 통합 버스에 연결되는 경우, 임시 큐가 버스에 작성되고 임시 큐의 이름을 구성하는 데 사용되는 접두부는 XMSC_WPM_TEMP_Q_PREFIX 특성에서 지정됩니다.

서비스 통합 버스에 연결된 애플리케이션이 임시 토픽을 작성하면 임시 토픽의 이름을 구성하는 데 사용되는 접두부는 XMSC_WPM_TEMP_TOPIC_PREFIX 특성에서 지정됩니다.

**관련 개념**

토픽 URI(Uniform Resource Identifier)
토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

큐 URI(Uniform Resource Identifier)
큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

목적지 와일드 카드
XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

**관련 참조**

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 목적지 와일드 카드

XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

설계는 다음과 같습니다.

| 연결 유형 | 와일드 카드 설계 | 설명 |
| --- | --- | --- |
| WebSphere MQ 큐 관리자 | *<br>?<br>% | 0개 이상의 문자<br>1개의 문자<br>이스케이프 문자 |
| 브로커에 대한 실시간 연결 | #<br>+ | 다중 레벨 일치<br>단일 레벨 일치 |
| WebSphere 서비스 통합 버스 | *<br>//<br>//. | 계층 구조의 한 레벨에서 문자 일치<br>0개 이상의 레벨 일치<br>Topic 표현식의 끝에서 0개 이상의 레벨 일치 |

**관련 개념**

토픽 URI(Uniform Resource Identifier)
토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

큐 URI(Uniform Resource Identifier)
큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

임시 목적지
XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

**관련 참조**

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 메시지 생성자

XMS에서 메시지 생성자는 유효한 목적지가 있는 상태로 또는 연관된 목적지가 없는 상태로 작성될 수 있습니다. 목적지가 널인 메시지 생성자를 작성하는 경우 메시지를 전송할 때 유효한 목적지를 지정해야 합니다.

### 연관된 목적지가 없는 메시지 생성자

XMS .NET에서 널 대상을 사용하여 메시지 생성자를 작성할 수 있습니다.

.NET API를 사용할 때 연관된 대상이 없는 메시지 생성자를 작성하려면, NULL은 매개변수로 ISession 오브젝트의 `CreateProducer()` 메소드에 전달되어야 합니다 (예: `session.CreateProducer(null)`). 그러나 메시지가 전송될 때 올바른 대상을 지정해야 합니다.

### 연관된 목적지가 있는 메시지 생성자

이 시나리오에서 메시지 생성자는 유효한 목적지를 사용하여 작성됩니다. 전송 조작 중 목적지를 지정할 필요가 없습니다.

## 메시지 이용자

메시지 이용자는 지속 가능/비지속 가능 구독자와 동기/비동기 메시지 이용자로 구분될 수 있습니다.

### 지속 가능 구독자

지속 가능 구독자는 구독자가 비활성인 동안 발행된 메시지를 포함하여 토픽에 대해 발행된 모든 메시지를 수신하는 메시지 이용자입니다.

이 절의 정보는 애플리케이션이 WebSphere MQ 큐 관리자 또는 WebSphere 서비스 통합 버스에 연결하는 경우에만 관련됩니다. 정보는 브로커에 대한 실시간 연결과 관련이 없습니다.

토픽에 대한 지속 가능 구독자를 작성하기 위해 애플리케이션은 지속 가능 구독자를 식별하는 이름과 토픽을 표시하는 Destination 오브젝트를 매개변수로 지정하여 Session 오브젝트의 Create Durable Subscriber 메소드를 호출합니다. 애플리케이션은 메시지 선택자를 사용하거나 메시지 선택자를 사용하지 않고 지속 가능 구독자를 작성할 수 있으며 지속 가능 구독자가 자체 연결에 의해 발행된 메시지를 수신할지 여부를 지정할 수 있습니다.

지속 가능 구독자를 작성하는 데 사용되는 세션에는 연관된 클라이언트 ID가 있어야 합니다. 클라이언트 ID는 세션 작성에 사용되는 연결과 연관된 ID와 동일하며 21 페이지의 『ConnectionFactories 오브젝트 및 Connection 오브젝트』에 설명되어 있는 대로 지정됩니다.

지속 가능 구독을 식별하는 이름은 클라이언트 ID 내에서 고유해야 하므로 클라이언트 ID는 지속 가능 구독의 고유한 전체 ID 일부를 구성합니다. 메시징 서버는 지속 가능 구독의 레코드를 유지보수하며 주제에 발행된 모든 메시지가 지속 가능 구독자에 의해 수신확인되거나 만료될 때까지 보존합니다.

메시징 서버는 지속 가능 구독자가 닫힌 후에도 계속 지속 가능 구독의 레코드를 유지보수합니다. 이미 작성된 지속 가능 구독을 재사용하려면 애플리케이션이 지속 가능 구독과 연관된 같은 구독 이름을 지정하고 같은 클라이언트 ID가 있는 세션을 사용하여 지속 가능 구독자를 작성해야 합니다. 한 번에 한 세션에만 특정 지속 가능 구독의 지속 가능 구독자가 있을 수 있습니다.

지속 가능한 구독의 범위는 구독 레코드를 유지보수하는 메시징 서버입니다. 다른 메시징 서버에 연결된 두 애플리케이션이 각각 같은 구독 이름과 클라이언트 ID를 사용하여 지속 가능 구독자를 작성하면 완전히 독립된 두 개의 지속 가능 구독이 작성됩니다.

지속 가능 구독을 삭제하기 위해 애플리케이션은 지속 가능 구독을 식별하는 이름을 매개변수로 지정하여 Session 오브젝트의 Unsubscribe 메소드를 호출합니다. 세션과 연관된 클라이언트 ID는 지속 가능 구독과 연관된 ID와 같아야 합니다. 메시징 서버는 유지보수 중인 지속 가능 구독의 레코드를 삭제하고 지속 가능 구독자에게 추가 메시지를 전송하지 않습니다.

기존 구독을 변경하기 위해 애플리케이션은 동일한 구독 이름과 클라이언트 ID를 사용하고 다른 토픽이나 메시지 선택자(또는 둘 다)를 지정하여 지속 가능 구독자를 작성할 수 있습니다. 지속 가능 구독을 변경하는 것은 구독을 삭제하고 새로 작성하는 것과 같습니다.

WebSphere MQ V7.0 큐 관리자에 연결하는 애플리케이션의 경우, XMS는 구독자 큐를 관리합니다. 그러므로 애플리케이션은 구독자 큐를 지정할 필요가 없습니다. XMS는 구독자 큐가 지정된 경우 이를 무시합니다.

그러나 WebSphere MQ 6.0 큐 관리자에 연결하는 애플리케이션의 경우, 각 지속 가능한 구독자에는 지정된 구독자 큐가 있어야 합니다. 토픽에 대한 구독자 큐의 이름을 지정하려면 토픽을 표시하는 Destination 오브젝트의 XMSC_WMQ_DUR_SUBQ 특성을 설정하십시오. 기본 구독자 큐는 SYSTEM.JMS.D.SUBSCRIBER.QUEUE입니다.

WebSphere MQ V6.0 큐 관리자에 연결하는 지속 가능한 구독자가 단일 구독자 큐를 공유할 수 있거나, 지속 가능한 각 구독자가 각각의 독점적인 구독자 큐에서 메시지를 검색할 수 있습니다. 사용자 애플리케이션에 맞게 채택하기 위한 접근 방법에 대한 토론을 보려면 *Java*를 사용하는 *WebSphere MQ*를 참조하십시오.

지속 가능 구독에 대한 구독자 큐는 변경할 수 없습니다. 구독자 큐를 변경하는 유일한 방법은 구독을 삭제하고 새로 작성하는 것입니다.

서비스 통합 버스에 연결된 애플리케이션의 경우 각 지속 가능 구독자에는 지정된 지속 가능 구독 홈이 있어야 합니다. 동일한 연결을 사용하는 모든 지속 가능 등록자에 대한 지속 가능 구독 홈을 지정하려면 연결을 작성하는 데 사용되는 ConnectionFactory 오브젝트의 XMSC_WPM_DUR_SUB_HOME 특성을 설정하십시오. 각 토픽에 대한 지속 가능 구독 홈을 지정하려면 토픽을 표시하는 Destination 오브젝트의 XMSC_WPM_DUR_SUB_HOME 특성을 설정하십시오. 연결에 대한 지속 가능 구독 홈을 지정해야 애플리케이션에서 연결을 사용하는 지속 가능 구독자를 작성할 수 있습니다. 목적지에 지정한 값은 연결에 지정한 값을 대체합니다.

## 지속 불가능한 구독자

지속 불가능한 구독자는 구독자가 활성인 동안 발행되는 메시지만을 수신하는 메시지 이용자입니다. 구독자가 비활성 상태일 때 전달되는 메시지는 유실됩니다.

이 절의 정보는 WebSphere MQ V6.0 큐 관리자를 통해 발행/구독 메시징을 사용하는 경우에만 관련이 있습니다.

연결 닫기 전 또는 연결 닫는 도중 이용자 오브젝트가 삭제되지 않으면 메시지는 더 이상 활성화되지 않는 구독자의 브로커 큐에서 그대로 유지될 수 있습니다.

이러한 경우에는 JMS의 WebSphere MQ 클래스와 함께 제공된 정리 유틸리티를 사용하여 큐에서 이러한 메시지를 지울 수 있습니다. 이 유틸리티를 사용하는 방법에 대한 자세한 내용은 *Java*를 사용하는 *WebSphere MQ*에 제공됩니다. 이 큐에 남는 메시지가 많은 경우 구독자 큐의 큐 용량을 늘려야 하는 경우도 있습니다.

## 동기 메시지 이용자

동기 메시지 이용자는 동시에 큐로부터 메시지를 수신합니다.

동기 메시지 이용자는 한 번에 하나의 메시지를 수신합니다. `Receive(wait interval)` 메소드가 사용되는 경우 호출은 지정된 기간(밀리초) 동안 또는 메시지 이용자가 닫을 때까지 메시지를 대기합니다.

ReceiveNoWait() 메소드가 사용되는 경우 동기 메시지 이용자는 지연 없이 메시지를 수신합니다. 다음 메시지가 사용 가능한 경우 즉시 수신됩니다. 그렇지 않으면 널 Message 오브젝트에 대한 포인터가 리턴됩니다.

## 비동기 메시지 이용자

비동기적 메시지 이용자는 비동기적으로 큐로부터 메시지를 수신합니다. 애플리케이션에 의해 등록된 메시지 리스너는 큐에서 새 메시지를 사용할 수 있을 때마다 호출됩니다.

## *XMS*의 변조 메시지

변조 메시지는 수신 MDB 애플리케이션이 처리할 수 없는 메시지입니다. 변조 메시지를 발견하면, XMS MessageConsumer 오브젝트는 두 개의 큐 특성(BOQUEUE 및 BOTHRESH)에 따라 이를 요청할 수 있습니다.

일부 환경에서 MDB에 전달된 메시지는 IBM WebSphere MQ 큐에 롤백될 수 있습니다. 예를 들어 메시지가 나중에 롤백되는 작업 단위 내에서 전달되는 경우 발생할 수 있습니다. 롤백되는 메시지는 일반적으로 다시 전달되지만, 잘못 형식화된 메시지로 인해 반복적으로 MDB가 실패하므로 전달될 수 없습니다. 이러한 메시지는 변조 메시지라고 합니다. 추가 조사를 위해 변조 메시지가 다른 큐로 자동 전송되거나 버려질 수 있도록 IBM WebSphere MQ를 구성할 수 있습니다. 이런 방식으로 IBM WebSphere MQ를 구성하는 방법에 대한 정보를 보려면 ASF에서 변조 메시지 핸들링을 참조하십시오.

종종 잘못된 형식의 메시지가 큐에 도착합니다. 이 컨텍스트에서, 잘못된 형식은 수신 애플리케이션이 메시지를 올바르게 처리할 수 없음을 의미합니다. 해당 메시지로 인해 수신 애플리케이션이 실패할 수 있으며, 이 잘못된 형식의 메시지를 백아웃할 수 있습니다. 그리고 메시지는 반복적으로 입력 큐에 전달되고 반복적으로 애플리케이션에 의해 백아웃될 수 있습니다. 이러한 메시지를 변조 메시지라고 합니다. XMS MessageConsumer 오브젝트는 변조 메시지를 감지하고 이를 대체 목적지로 다시 라우팅합니다.

IBM WebSphere MQ 큐 관리자는 각 메시지가 백아웃한 횟수의 레코드를 유지합니다. 이 숫자가 구성 가능한 임계값에 도달한 경우, 메시지 이용자는 메시지를 이름 지정된 백아웃 큐에 리큐잉합니다. 이 리큐잉이 어떤 이유로든 실패하는 경우, 메시지가 입력 큐에서 제거되며 데드-레터 큐에 리큐잉되거나 제거됩니다.

XMS ConnectionConsumer 오브젝트는 동일한 방식으로 변조 메시지를 처리하고 동일한 큐 특성을 사용합니다. 다수의 연결 이용자가 동일한 큐를 모니터링하는 경우, 리큐잉이 발생하기 전에 임계값을 초과하는 횟수로 변조 메시지가 애플리케이션에 전달될 수 있습니다. 이 작동은 개별 연결 이용자 모니터가 변조 메시지를 큐잉하고 리큐잉하는 방식 때문입니다.

임계값과 백아웃 큐의 이름은 IBM WebSphere MQ 큐의 속성입니다. 속성의 이름은 BackoutThreshold 및 BackoutRequeueQName입니다. 이러한 속성이 적용되는 큐는 다음과 같습니다.

- 포인트-투-포인트 메시징의 경우, 이는 기본 로컬 큐입니다. 이는 메시지 이용자 및 연결 이용자가 큐 알리어스를 사용할 때 중요합니다.
- IBM WebSphere MQ 메시징 제공자 정상 모드에서의 발행/구독 메시징의 경우, 이는 토픽의 관리 큐가 작성되는 모델 큐입니다.
- IBM WebSphere MQ 메시징 제공자 마이그레이션 모드의 발행/구독 메시징의 경우, 이는 TopicConnectionFactory 오브젝트에 정의된 CCSUB 큐이거나 Topic 오브젝트에 정의된 CCDSUB 큐입니다.

BackoutThreshold 및 BackoutRequeueQName 속성을 설정하려면 다음의 MQSC 명령을 실행하십시오.

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

발행/구독 메시징의 경우, 시스템이 각 구독에 대해 동적 큐를 작성할 때 이러한 속성 값은 JMS용 WebSphere MQ 클래스 모델 큐(SYSTEM.JMS.MODEL.QUEUE)에서 확보됩니다. 이러한 설정을 대체하려면 다음을 사용하십시오.

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

백아웃 임계값이 0인 경우에는 변조 메시지 핸들링이 사용되지 않으며 변조 메시지가 입력 큐에 남아 있습니다. 그렇지 않으면, 백아웃 계수가 임계값에 도달할 때 메시지가 이름 지정된 백아웃 큐에 송신됩니다.

백아웃 수가 임계값에 도달하지만 메시지를 백아웃 큐로 이동시킬 수 없으면 메시지는 데드-레터 큐로 전송되거나 메시지가 비지속 메시지인 경우 제거됩니다.

이러한 상황은 백아웃 큐가 정의되지 않았거나 MessageConsumer 오브젝트가 메시지를 백아웃 큐에 송신할 수 없는 경우에 발생합니다.

## 변조 메시지 핸들링을 수행하도록 시스템 구성

XMS .NET에서 **BOTHRESH** 및 **BOQNAME** 속성을 조회할 때 사용할 큐는 수행하는 메시징 스타일에 따라 달라집니다.

- 포인트-투-포인트 메시징의 경우, 이는 기본 로컬 큐입니다. XMS .NET 애플리케이션이 알리어스 큐 또는 클러스터 큐의 메시지를 이용할 경우에 중요합니다.
- 발행/구독 메시징의 경우, 애플리케이션의 메시지를 보관하기 위해 관리 큐가 작성됩니다. XMS .NET에서는 관리 큐를 조회하여 **BOTHRESH** 및 **BOQNAME** 속성의 값을 판별합니다.

  관리 큐는 애플리케이션이 구독하는 토픽 오브젝트와 연관된 모델 큐에서 작성되며, **BOTHRESH** 및 **BOQNAME** 속성의 값을 모델 큐로부터 상속받습니다. 수신 애플리케이션이 지속 가능 구독 또는 지속 불가능 구독을 제공했는지 여부에 따라 사용되는 모델 큐가 달라집니다.

  - 지속 가능 구독에 사용되는 모델 큐는 토픽의 **MDURMDL** 속성에 지정됩니다. 이 속성의 기본값은 `SYSTEM.DURABLE.MODEL.QUEUE`입니다.
  - 지속 불가능 구독에 사용되는 모델 큐는 **MNDURMDL** 속성에 지정됩니다. **MNDURMDL** 속성의 기본값은 `SYSTEM.NDURABLE.MODEL.QUEUE`입니다.

**BOTHRESH** 및 **BOQNAME** 속성을 조회할 때 XMS .NET에서는 다음을 수행합니다.

- 로컬 큐 또는 알리어스 큐의 대상 큐를 엽니다.
- **BOTHRESH** 및 **BOQNAME** 속성을 조회합니다.
- 로컬 큐 또는 알리어스 큐의 대상 큐를 닫습니다.

로컬 큐 또는 알리어스 큐의 대상 큐를 열 때 사용되는 열기 옵션은 사용 중인 IBM MQ의 버전에 따라 다릅니다.

- 버전 8.0.0, 수정팩 13 및 이전 버전의 경우 알리어스 큐에 대한 로컬 큐 또는 대상 큐가 클러스터 큐인 경우 XMS .NET은 `MQOO_INPUT_AS_Q_DEF`, `MQOO_INQUIRE` 및 `MQOO_FAIL_IF_QUIESCING` 옵션을 사용하여 큐를 엽니다. 즉, 수신 애플리케이션을 실행하는 사용자가 클러스터 큐의 로컬 인스턴스를 조회하고 이 인스턴스에 액세스할 수 있는 권한을 갖고 있어야 합니다.

  XMS .NET에서는 열기 옵션 `MQOO_INQUIRE` 및 `MQOO_FAIL_IF_QUIESCING`을 사용하여 로컬 큐의 다른 모든 유형을 엽니다. XMS .NET에서 속성 값을 조회하려면 수신 애플리케이션을 실행하는 사용자가 로컬 큐에 대한 조회 액세스 권한을 갖고 있습니다.
- V 8.0.0.14 버전 8.0.0, 수정팩 14에서 XMS .NET을 사용하는 경우 수신 애플리케이션을 실행 중인 사용자에게 큐의 유형에 관계 없이 로컬 큐에 대한 조회 액세스 권한이 있어야 합니다.

변조 메시지를 백아웃 큐 또는 큐 관리자의 데드-레터 큐로 이동시키려면 애플리케이션을 실행하는 사용자에게 `put` 및 `passall` 권한을 부여해야 합니다.

*ASF*에서 변조 메시지 핸들링
ASF(Application Server Facilities)를 사용할 때는 MessageConsumer가 아닌 ConnectionConsumer가 변조 메시지를 처리합니다. ConnectionConsumer는 큐의 BackoutThreshold 및 BackoutRequeueQName 특성에 따라 메시지를 리큐잉합니다.

애플리케이션이 ConnectionConsumers를 사용하는 경우, 메시지가 백아웃되는 환경은 애플리케이션 서버가 제공하는 세션에 달려 있습니다.

- 세션이 AUTO_ACKNOWLEDGE 또는 DUPS_OK_ACKNOWLEDGE로 트랜잭션되지 않은 경우, 메시지는 시스템 오류 이후에만 또는 애플리케이션이 예상치 못하게 종료된 경우에 백아웃됩니다.
- 세션이 CLIENT_ACKNOWLEDGE로 트랜잭션되지 않는 경우, 수신확인되지 않은 메시지는 `Session.recover()`를 호출하는 애플리케이션 서버에 의해 백아웃될 수 있습니다.

  일반적으로 MessageListener 또는 애플리케이션 서버의 클라이언트 구현에서는 `Message.acknowledge()`를 호출합니다. `Message.acknowledge()`는 지금까지 세션에 전달된 모든 메시지를 수신확인합니다.
- 세션이 트랜잭션되면, 수신확인되지 않은 메시지는 `Session.rollback()`을 호출하는 애플리케이션 서버에 의해 백아웃될 수 있습니다.

## 큐 브라우저

애플리케이션은 큐 브라우저를 사용하여 큐의 메시지를 제거하지 않고 찾아봅니다.

큐 브라우저를 작성하기 위해, 애플리케이션은 찾을 큐를 식별하는 Destination 오브젝트를 매개변수로서 지정하는 ISession 오브젝트의 Create Queue Browser 메소드를 호출합니다. 애플리케이션은 메시지 선택자를 사용하거나 메시지 선택자 없이 큐 브라우저를 작성할 수 있습니다.

큐 브라우저를 작성한 후 애플리케이션은 IQueueBrowser 오브젝트의 GetEnumerator 메소드를 호출하여 큐에서 메시지 목록을 가져올 수 있습니다. 이 메소드는 Message 오브젝트의 목록을 캡슐화하는 열거자를 리턴합니다. 목록에 있는 Message 오브젝트의 순서는 큐에서 메시지를 검색하는 순서와 같습니다. 애플리케이션은 열거자를 사용하여 각 메시지를 교대로 찾아볼 수 있습니다.

열거자는 메시지가 큐에 넣어지고 큐에서 제거될 때 동적으로 업데이트됩니다. 애플리케이션이 큐에서 다음 메시지를 찾기 위해 IEnumerator.MoveNext()를 호출할 때마다 메시지는 큐의 현재 컨텐츠를 반영합니다.

애플리케이션은 지정된 큐 브라우저에 대해 GetEnumerator 메소드를 두 번 이상 호출할 수 있습니다. 각 호출에서는 새로운 열거자를 리턴합니다. 그러므로 애플리케이션은 둘 이상의 열거자를 사용하여 큐의 메시지를 찾아보고 큐 내에 여러 위치를 유지보수할 수 있습니다.

애플리케이션은 큐 브라우저를 사용하여 큐에서 제거할 수 있는 적합한 메시지를 검색한 후 메시지 선택자와 함께 메시지 이용자를 사용하여 메시지를 제거할 수 있습니다. 메시지 선택자는 JMSMessageID 헤더 필드의 값에 따라 메시지를 선택할 수 있습니다. 이 헤더 필드 및 기타 JMS 메시지 헤더 필드에 대한 정보는 64 페이지의 『XMS 메시지의 헤더 필드』의 내용을 참조하십시오.

## 요청자

애플리케이션은 요청자를 사용하여 요청 메시지를 보낸 후 응답을 대기 및 수신합니다.

대부분의 메시징 애플리케이션은 요청 메시지를 보낸 후 응답을 대기하는 알고리즘을 기반으로 합니다. XMS는 이런 스타일의 애플리케이션 개발에 도움이 되는 Requestor 클래스를 제공합니다.

요청자를 작성하기 위해 애플리케이션은 요청 메시지를 보낼 위치를 식별하는 Destination 및 Session 오브젝트를 매개변수로 지정하여 Requestor 클래스의 Create Requestor 생성자를 호출합니다. 세션은 트랜잭션되지 않아야 하고 XMSC_CLIENT_ACKNOWLEDGE 수신확인 모드가 없어야 합니다. 생성자는 응답 메시지를 보낼 임시 큐 또는 토픽을 자동으로 작성합니다.

요청자가 작성되면 애플리케이션은 Requestor 오브젝트의 Request 메소드를 호출하여 요청 메시지를 보낸 후 요청 메시지를 수신하는 애플리케이션으로부터 응답을 대기 및 수신합니다. 호출은 응답을 수신하거나 세션이 끝나거나 빨리 발생하는 이벤트를 대기합니다. 요청자에서는 각 요청 메시지에 대해 한 개의 응답만 필요합니다.

애플리케이션이 요청자를 닫으면 임시 큐 또는 토픽이 삭제됩니다. 그러나 연관된 세션은 닫히지 않습니다.

## 오브젝트 삭제

애플리케이션이 작성된 XMS 오브젝트를 삭제하는 경우 XMS는 오브젝트에 할당된 내부 자원을 릴리스합니다.

애플리케이션이 XMS 오브젝트를 작성하는 경우, XMS는 오브젝트에 메모리와 다른 내부 자원을 할당합니다. XMS는 XMS가 내부 자원을 릴리스할 때 오브젝트의 닫기 또는 삭제 메소드를 호출하여 애플리케이션이 오브젝트를 명시적으로 삭제할 때까지 이러한 내부 자원을 보존합니다. 애플리케이션이 이미 삭제된 오브젝트를 삭제하려고 하면 호출이 무시됩니다.

애플리케이션이 Connection 또는 Session 오브젝트를 삭제하면 XMS는 연관된 특정 오브젝트를 자동으로 삭제하고 내부 자원을 릴리스합니다. 이는 Connection 또는 Session 오브젝트에서 작성한 오브젝트이며 오브젝트에 연동되어 작동합니다. 이러한 오브젝트는 36 페이지의 표 4에 표시되어 있습니다.

**참고:** 애플리케이션이 종속 세션과의 연결을 닫으면 해당 세션에 종속되는 모든 오브젝트도 삭제됩니다. Connection 또는 Session 오브젝트에만 종속 오브젝트가 있을 수 있습니다.

표 *4.* 자동으로 삭제되는 오브젝트

| **삭제된 오브젝트** | **방법** | **자동으로 삭제되는 종속 오브젝트** |
|---|---|---|
| 연결 | Close Connection | ConnectionMetaData 및 Session 오브젝트 |
| 세션 | Close Session | MessageConsumer, MessageProducer, QueueBrowser 및 Requestor 오브젝트 |

## XMS를 통한 관리 WebSphere MQ XA 트랜잭션

관리 WebSphere MQ XA 트랜잭션은 XMS를 통해 사용할 수 있습니다.

XMS를 통해 XA 트랜잭션을 사용하려면 트랜잭트된 세션을 작성해야 합니다. XA 트랜잭션이 사용 중인 경우, 트랜잭션 제어는 DTC(Distributed Transaction Coordinator) 글로벌 트랜잭션을 통하고 XMS 세션은 사용하지 않습니다. XA 트랜잭션을 사용하는 경우, `Session.commit` 또는 `Session.rollback`을 XMS 세션에서 발행할 수 없습니다. 대신, `Transscope.Commit` 또는 `Transscope.Rollback` DTC 메소드 커미트를 사용하거나 트랜잭션을 롤백하십시오. 세션이 XA 트랜잭션에 사용되는 경우, 세션을 사용하여 작성되는 생성자 또는 이용자는 XA 트랜잭션의 일부여야 합니다. 이들은 XA 트랜잭션 범위 이외의 조작에 대해서는 사용할 수 없습니다. 이들은 XA 트랜잭션 외부의 `Producer.send` 또는 `Consumer.receive`와 같은 조작에 사용할 수 없습니다.

`IllegalStateException` 예외 오브젝트는 다음 경우에 처리됩니다.

- `Session.commit` 또는 `Session.rollback`에 대해 XA 트랜잭션 세션이 사용됩니다.
- XA 트랜잭션 세션에서 사용되었던 생성자 또는 이용자는 XA 트랜잭션 범위 외부에서 사용됩니다.

XA 트랜잭션은 비동기 이용자에서 지원되지 않습니다.

**참고:**

1. XA 트랜잭션을 커미트하기 전에 `Producer`, `Consumer`, `Session` 또는 `Connection` 오브젝트에서 닫기가 발행될 수 있습니다. 이런 경우 트랜잭션의 메시지는 롤백됩니다. 마찬가지로 XA 트랜잭션을 커미트하기 전에 연결이 끊어지면 트랜잭션에 있는 모든 메시지는 롤백됩니다. `Producer` 오브젝트의 경우 롤백은 메시지가 큐에 넣어지지 않음을 의미합니다. `Consumer` 오브젝트의 경우 롤백은 메시지가 큐에 남아 있음을 의미합니다.
2. `Producer` 오브젝트가 `TimeToLive` 상태의 메시지를 `TransactionScope`에 넣고 시간이 경과하면 `commit`를 발행하는 경우, `commit`를 발행하기 전에 메시지가 만료될 수 있습니다. 이런 경우 메시지는 `Consumer` 오브젝트에서 사용할 수 없게 됩니다.
3. `Session` 오브젝트는 스레드에서 지원되지 않습니다. 스레드에서 공유되는 `Session` 오브젝트에서의 트랜잭션 사용은 지원되지 않습니다.

## XMS 기본 유형

XMS에서는 동일한 8개의 Java 기본 유형(byte, short, int, long, float, double, char 및 boolean)을 제공합니다. 이를 사용하여 데이터 손실이나 손상 없이 XMS와 JMS 간에 메시지를 교환할 수 있습니다.

37 페이지의 표 5에는 Java에 상응하는 데이터 유형, 크기 및 각 XMS 기본 유형의 최소값 및 최대값이 표시됩니다.

표 *5. XMS* 데이터 유형 및 *Java* 등가물

| **XMS 데이터 유형** | **호환 가능 Java 데이터 유형** | **크기** | **최소값** | **최대값** |
|---|---|---|---|---|
| System.Boolean | 부울 | 32비트 | false | true |
| System.SBYTE | byte | 8비트 | $-2^7$(-128 ) | $2^7$-1(127) |
| System.BYTE | byte | 8비트 | $-2^7$(-128 ) | $2^7$-1(127) |
| System.CHAR | byte | 8비트 | $-2^7$(-128 ) | $2^7$-1(127) |
| System.Int16 | 쇼트 | 16비트 | $-2^{15}$(-32768) | $2^{15}$-1(32767) |
| System.Int32 | int | 32비트 | $-2^{31}$(-2147483648) | $2^{31}$-1(2147483647) |
| System.Int64 | 롱 | 64비트 | $-2^{63}$(-9223372036854775808) | $2^{63}$-1(9223372036854775807) |
| System.Single | 부동 소수점 | 32비트 | -3.402823E-38(7자리 정밀도까지) | 3.402823E+38(7자리 정밀도까지) |
| System.Double | 실수 (double) | 64비트 | -1.79769313486231E-308(15자리 정밀도까지) | 1.79769313486231E+308(15자리 정밀도까지) |

**관련 개념**

오브젝트의 속성 및 특성

XMS 오브젝트에는 여러 가지 방법으로 구현되는 오브젝트 특성인 속성과 특성을 포함할 수 있습니다.

한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환

애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형

XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

## 한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환

애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

오브젝트 특성에는 이름과 값이 있으며 값에는 연관된 데이터 유형이 있습니다. 특성 값은 특성 유형이라고도 합니다.

애플리케이션은 PropertyContext 클래스의 메소드를 사용하여 오브젝트의 특성을 가져오고 설정합니다. 애플리케이션은 특성 값을 가져오기 위해 특성 유형에 적합한 메소드를 호출합니다. 예를 들어 정수 특성 값을 가져오기 위해 애플리케이션은 일반적으로 GetIntProperty 메소드를 호출합니다.

그러나 애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 예를 들어 정수 특성 값을 가져오기 위해 애플리케이션은 GetStringProperty 메소드를 호출할 수 있습니다. 이는 특성 값을 문자열로 리턴합니다. XMS에서 지원하는 변환은 37 페이지의 표 6에 표시되어 있습니다.

표 *6.* 특성 유형에서 다른 데이터 유형으로 지원되는 변환

| **등록 정보 유형** | **지원되는 대상 데이터 유형** |
|---|---|
| System.String | System.Boolean, System.Double, System.Float, System.Int32, System.Int64, System.SByte, System.Int16 |

| 표 6. 특성 유형에서 다른 데이터 유형으로 지원되는 변환 (계속) | |
|---|---|
| **등록 정보 유형** | **지원되는 대상 데이터 유형** |
| 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 |

지원되는 변환에는 다음 일반 규칙이 적용됩니다.

- 변환 도중 데이터가 손실되지 않으면 숫자 등록 정보 값을 다른 데이터 유형으로 변환할 수 있습니다. 예를 들어 System.Int32 데이터 유형의 특성 값은 System.Int64 데이터 유형의 값으로 변환될 수 있지만, 이는 System.Int16 데이터 유형의 값으로 변환될 수 없습니다.
- 데이터 유형의 특성 값은 문자열로 변환될 수 있습니다.
- 문자열이 변환에 맞게 형식화된 경우 문자열 특성 값은 다른 데이터 유형으로 변환될 수 있습니다. 애플리케이션이 올바로 형식화되지 않은 문자열 특성 값을 변환하려고 하면 XMS가 오류를 리턴할 수 있습니다.
- 애플리케이션이 지원되지 않는 변환을 시도하면 XMS가 오류를 리턴할 수 있습니다.

다음 규칙은 특성 값이 하나의 데이터 유형에서 다른 데이터 유형으로 변환되는 경우에 적용됩니다.

- Boolean 특성 값을 문자열로 변환하는 경우, True 값은 "true" 문자열로 변환되고, False 값은 "false" 문자열로 변환됩니다.
- System.SByte를 포함하여 Boolean 특성 값을 숫자 데이터 유형으로 변환하는 경우 True 값은 1로 변환되고 False 값은 0으로 변환됩니다.
- 문자열 특성 값을 Boolean 값으로 변환하는 경우, "true" 문자열(대소문자 구분 없음) 또는 "1"은 True로 변환되고 "false"문자열(대소문자 구분 없음) 또는 "0"은 False로 변환됩니다. 기타 모든 문자열은 변환될 수 없습니다.
- 문자열 특성 값을 System.Int32, System.Int64, System.SByte 또는 System.Int16 데이터 유형의 값으로 변환하는 경우, 문자열은 다음 형식이어야 합니다.

  [ *blanks* ][ *sign* ] *digits*

  문자열 컴포넌트는 다음과 같이 정의됩니다.

  **공백**
  선택적 선두 공백 문자입니다.

  ***sign***
  선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

  ***digits***
  연속적인 순서의 숫자 문자(0-9). 하나 이상의 숫자가 있어야 합니다.

  문자열은 숫자의 연속 뒤에 숫자가 아닌 다른 문자를 포함할 수 있지만 이러한 문자의 첫 번째에 이르면 변환을 중지합니다. 문자열은 10진수 정수를 표시한다고 가정됩니다.

  XMS에서는 문자열이 올바로 형식화되지 않으면 오류를 리턴할 수 있습니다.
- 문자열 특성 값을 System.Double 또는 System.Float 데이터 유형의 값으로 변환하는 경우, 문자열의 형식은 다음과 같아야 합니다.

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

문자열 컴포넌트는 다음과 같이 정의됩니다.

**공백**
(선택적) 선행 공백 문자.

***sign***
(선택적) 더하기 부호(+) 또는 빼기 부호(-) 문자.

***digits***
연속적인 순서의 숫자 문자(0-9). *digits* 또는 *d_digits*에는 하나 이상의 숫자가 있어야 합니다.

***point***
(선택사항) 소수점(.).

***d_digits***
연속적인 순서의 숫자 문자(0-9). *digits* 또는 *d_digits*에는 하나 이상의 숫자가 있어야 합니다.

***e_char***
지수 문자(*E* 또는 *e*)입니다.

***e_sign***
(선택적) 지수의 더하기 부호(+) 또는 빼기 부호(-) 문자.

***e_digits***
지수에 대한 연속적인 순서의 숫자 문자(0-9). 문자열에 지수 문자가 포함된 경우 하나 이상의 숫자가 있어야 합니다.

문자열은 숫자의 연속 또는 지수를 나타내는 선택적 문자 뒤에 숫자가 아닌 다른 문자를 포함할 수 있지만 이러한 문자의 첫 번째에 이르면 변환을 중지합니다. 문자열은 10제곱인 지수의 10진수 부동 소수점 숫자를 표시한다고 가정됩니다.

XMS에서는 문자열이 올바로 형식화되지 않으면 오류를 리턴할 수 있습니다.

- System.SByte 데이터 유형의 특성 값을 포함하는 숫자 특성 값을 문자열로 변환하는 경우, 값은 10진수로서의 값 문자열 표현으로 변환됩니다. 해당 값에 대한 ASCII 문자를 포함하는 문자열로 변환되지 않습니다. 예를 들어, 정수 65는 문자열 "A"가 아닌 문자열 "65"로 변환됩니다.
- 바이트 배열 특성 값을 문자열로 변환하면 각 바이트는 바이트를 표시하는 두 개의 16진 문자로 변환됩니다. 예를 들어 바이트 배열 {0xF1, 0x12, 0x00, 0xFF}은 문자열 "F11200FF"로 변환됩니다.

특성 유형에서 다른 데이터 유형으로의 변환은 Property와 PropertyContext 클래스의 메소드에 의해 지원됩니다.

**관련 개념**
오브젝트의 속성 및 특성
XMS 오브젝트에는 여러 가지 방법으로 구현되는 오브젝트 특성인 속성과 특성을 포함할 수 있습니다.

XMS 기본 유형
XMS에서는 동일한 8개의 Java 기본 유형(byte, short, int, long, float, double, char 및 boolean)을 제공합니다. 이를 사용하여 데이터 손실이나 손상 없이 XMS와 JMS 간에 메시지를 교환할 수 있습니다.

**관련 참조**
맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

## 반복기

반복기는 목록에서 현재 위치를 유지하는 커서와 오브젝트의 목록을 캡슐화합니다. Message Service Client for C/C++에서 사용 가능한 것과 같이, 반복기 개념은 Message Service Client for .NET에서 IEnumerator 인터페이스를 사용하여 구현됩니다.

반복기가 작성되면 커서의 위치는 첫번째 오브젝트의 앞에 옵니다. 애플리케이션은 반복기를 사용하여 차례로 각 오브젝트를 검색합니다.

Message Service Client for C/C++의 Iterator 클래스는 Java의 Enumerator 클래스와 동일합니다.XMS .NET는 Java와 유사하며 IEnumerator 인터페이스를 사용합니다.

애플리케이션은 IEnumerator를 사용하여 다음 태스크를 수행할 수 있습니다.

- 메시지 특성 가져오기
- 맵 메시지의 본문에서 이름-값 쌍 가져오기
- 큐에서 메시지 찾아보기
- 연결에서 지원하는 JMS 정의 메시지 특성의 이름 가져오기

## 코드화 문자 세트 ID

XMS .NET에서 모든 문자열은 기본 .NET 문자열을 사용하여 전달됩니다. 이는 고정된 인코딩이므로 해석하는 데 추가 정보가 필요하지 않습니다. 그러므로 XMSC_CLIENT_CCSID 특성은 XMS .NET 애플리케이션에 필요하지 않습니다.

## XMS 오류 및 예외 코드

XMS는 다양한 오류 코드를 사용하여 실패를 표시합니다. 이러한 오류 코드는 릴리스마다 다를 수 있으므로 본 문서에 명시적으로 나열되지 않았습니다. XMS 예외 코드(XMS_X_... 형식)는 XMS 릴리스 사이에서 동일하게 유지되므로 이에 대해서만 설명합니다.

## 자체 애플리케이션 빌드

샘플 애플리케이션을 빌드하는 것처럼 자체 애플리케이션을 빌드할 수 있습니다.

19 페이지의 『.NET 샘플 애플리케이션 빌드』 에 설명된 대로 .NET 애플리케이션을 빌드하십시오. 이 태스크에는 자체 .NET 애플리케이션을 빌드하는 데 필요한 전제조건도 나열되어 있습니다. 자체 애플리케이션의 빌드 방법에 대한 추가 지침은 각 샘플 애플리케이션에 제공된 makefile을 사용하십시오.

**팁:** 실패할 경우 문제점 진단을 위해서는 제공된 기호로 애플리케이션을 컴파일하는 것이 좋습니다.

**관련 개념**

샘플 애플리케이션
샘플 애플리케이션은 각 API의 공통 기능에 대해 간략하게 설명합니다. 이를 사용하여 설치 및 메시징 서버 설정을 확인하고 자신만의 애플리케이션을 빌드하는 데 도움을 받을 수 있습니다.

**관련 참조**

.NET 인터페이스
이 절에서는 .NET 클래스 인터페이스와 해당 특성 및 메소드에 대해 설명합니다.

XMS 오브젝트 특성
이 장에서는 XMS에 의해 정의된 오브젝트 특성에 대해 설명합니다.

## *XMS*를 통한 자동 *WebSphere MQ* 클라이언트 재연결

WebSphere MQ V7.1 클라이언트 이상을 사용하는 동안 네트워크, 큐 관리자 또는 서버 실패 후에 메시지 제공자로서 자동으로 연결되도록 XMS 클라이언트를 구성합니다.

MQConnectionFactory 클래스의 WMQ_CONNECTION_NAME_LIST 및 WMQ_CLIENT_RECONNECT_OPTIONS 특성을 사용하여 자동으로 다시 연결하도록 클라이언트 연결을 구성하십시오. 클라이언트 자동 재연결은 연결 실패 후 또는 큐 관리자를 중지한 후 옵션으로서 클라이언트를 재연결합니다. 일부 클라이언트 애플리케이션의 디자인은 자동 연결에 적합하지 않습니다.

일단 연결이 설정되면 다시 연결 가능한 클라이언트 연결은 다시 연결 가능으로 자동 전환됩니다.

**참고:** 클라이언트 다시 연결 옵션, 클라이언트 다시 연결 제한시간 및 연결 이름 목록 특성은 CCDT(Client Channel Definitions Table)를 통해 설정되거나 mqclient.ini 파일을 통해 클라이언트 다시 연결을 사용으로 설정하여 설정할 수 있습니다.

**참고:** 다시 연결 특성이 CCDT에서와 함께 `ConnectionFactory` 오브젝트에서 설정되는 경우 서열 규칙은 다음과 같습니다. 연결 이름 목록 특성의 기본 값이 `ConnectionFactory` 오브젝트에서 설정되면 CCDT가 이에 우선합니다. 연결 이름 목록이 기본값으로 설정되지 않으면, `ConnectionFactory` 오브젝트에서 설정된 특성 값이 우선합니다. 연결 이름 목록의 기본값은 `localhost(1414)`입니다.

# XMS .NET 애플리케이션 작성

이 장에서는 XMS.NET 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

장에는 다음 절가 포함됩니다.



**관련 개념**

XMS 애플리케이션 작성
이 장에서는 XMS 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

**관련 참조**

.NET 인터페이스
이 절에서는 .NET 클래스 인터페이스와 해당 특성 및 메소드에 대해 설명합니다.

## .NET의 데이터 유형

XMS .NET에서는 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 및 System.Object를 지원합니다. XMS .NET의 데이터 유형은 XMS C++의 데이터 유형과 다릅니다. 해당하는 데이터 유형을 식별하기 위해 이 장을 사용할 수 있습니다.

다음 표에서 해당하는 XMS .NET 및 XMS C++ 데이터 유형을 표시하고 이 유형을 간략하게 설명합니다.

표 *7. XMS .NET* 및 *XMS C++*의 데이터 유형

| **XMS .NET 유형** | **XMS C++ 유형** | **설명** |
|---|---|---|
| System.SByte | xmsSBYTE<br>xmsINT8 | 부호가 있는 8비트 값 |
| System.Byte | xmsBYTE<br>xmsUINT8 | 부호가 없는 8비트 값 |
| System.Int16 | xmsINT16<br>xmsSHORT | 부호가 있는 16비트 값 |
| System.UInt16 | xmsUINT16<br>xmsUSHORT | 부호가 없는 16비트 값 |
| System.Int32 | xmsINT32<br>xmsINT | 부호가 있는 32비트 값 |

| 표 7. *XMS .NET* 및 *XMS C++*의 데이터 유형 *(계속)* | | |
|---|---|---|
| **XMS .NET 유형** | **XMS C++ 유형** | **설명** |
| System.UInt32 | xmsUINT32<br>xmsUINT | 부호가 없는 32비트 값 |
| System.Int64 | xmsLONG<br>xmsINT64 | 부호가 있는 64비트 값 |
| System.UInt64 | xmsULONG<br>xmsUINT64 | 부호가 없는 64비트 값 |
| System.Char | xmsCHAR16 | 부호가 없는 16비트 문자(.NET용 유니코드) |
| System.Single | xmsFLOAT | IEEE 32비트 Float |
| System.Double | xmsDOUBLE | IEEE 64비트 Float |
| System.Boolean | xmsBOOL | True/False 값 |
| 적용할 수 없음 | xmsCHAR | 부호가 있는 또는 부호가 없는 8비트 값 (부호의 유무는 플랫폼에 따라 결정됨) |
| System.Decimal | 적용할 수 없음 | 부호가 있는 96비트 정수 곱하기 $10^0$ - $10^{28}$ |
| System.Object | 적용할 수 없음 | 모든 유형의 기본 |
| System.String | 적용할 수 없음 | 문자열 유형 |

## .NET의 관리되는 운영 및 관리되지 않는 운영

관리 코드는 .NET 공통 언어 런타임 환경 내에서 배타적으로 실행되며 전적으로 해당 런타임에서 제공되는 서비스에 종속됩니다. 애플리케이션의 일부가 .NET 공용 언어 런타임 환경 외부에서 실행되거나 서비스를 호출하면 애플리케이션은 비관리로 분류됩니다.

관리 .NET 환경에서는 현재 특정 고급 기능을 지원할 수 없습니다.

완전히 관리되는 환경에서 현재 지원되지 않는 일부 기능이 애플리케이션에 필요한 경우 애플리케이션을 실제 변경할 필요 없이 애플리케이션이 비관리 환경을 사용하도록 변경할 수 있습니다. 그러나 이 선택사항이 지정되면 XMS 스택이 비관리 코드를 사용합니다.

### WebSphere MQ 큐 관리자에 대한 연결

WMQ_CM_CLIENT에 대한 관리 연결에서는 비TCP 통신 및 채널 압축을 지원하지 않습니다. 그러나 이러한 연결은 비관리 연결(WMQ_CM_CLIENT_UNMANAGED)을 사용하여 지원될 수 있습니다. 자세한 정보는 .NET 애플리케이션 개발을 참조하십시오.

비관리 환경의 관리 대상 오브젝트에서 연결 팩토리를 작성하는 경우 수동으로 연결 모드 값을 XMSC_WMQ_CM_CLIENT_UNMANAGED로 변경해야 합니다.

### WebSphere 서비스 통합 버스 메시징 엔진에 대한 연결

SSL 프로토콜(HTTPS 포함)을 사용해야 하는 WebSphere 서비스 통합 버스 메시징 엔진에 대한 연결은 현재 관리 코드로서 지원되지 않습니다.

**관련 참조**

XMSC_WMQ_CONNECTION_MODE

## .NET의 목적지

.NET에서 목적지는 프로토콜 유형에 따라 작성되며 목적지가 작성된 프로토콜 유형에서만 사용될 수 있습니다.

목적지 작성을 위한 두 가지 함수(토픽 함수와 큐 함수)가 제공됩니다.

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

이러한 함수는 API에서 다음과 같은 두 개의 오브젝트에 대해 사용 가능합니다.

- ISession
- XMSFactoryFactory

두 경우 모두에서 이러한 메소드는 매개변수를 포함하고 다음 형식의 URI 스타일 문자열을 승인할 수 있습니다.

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

또는 이러한 메소드에서 목적지 이름 즉, topic:// 또는 queue:// 접두부 그리고 매개변수가 없는 이름만 승인할 수도 있습니다.

따라서 URI 스타일 문자열은 다음과 같습니다.

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

다음과 같은 목적지 이름과 동일한 결과를 생성합니다.

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

WebSphere 서비스 통합 버스 JMS의 경우에는 토픽을 짧은 형식으로 지정할 수 있으며, 여기에 *topicname* 및 *topicspace*를 포함할 수 있지만 매개변수는 포함할 수 없습니다.

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

## .NET의 특성

.NET 애플리케이션은 PropertyContext 인터페이스의 메소드를 사용하여 오브젝트의 특성을 가져오고 설정합니다.

PropertyContext 인터페이스는 특성 가져오기 및 설정 메소드를 캡슐화합니다. 이러한 메소드는 다음 클래스에 의해 직접 또는 간접적으로 상속됩니다.

- BytesMessage
- Connection
- ConnectionFactory
- ConnectionMetaData
- 목적지
- MapMessage
- 메시지
- MessageConsumer
- MessageProducer
- ObjectMessage
- QueueBrowser
- 세션
- StreamMessage

- TextMessage

애플리케이션이 특성 값을 설정하면 새 값이 특성에 있는 이전 값을 대체합니다.

XMS 특성에 대한 자세한 정보는 167 페이지의 『XMS 오브젝트 특성』의 내용을 참조하십시오.

편리한 사용을 위해 XMS의 .NET 특성 이름 및 값이 XMSC 구조체의 공용 상수로 미리 정의되어 있습니다. 이러한 상수의 이름은 XMSC.*<constant>*양식입니다. 예를 들어, XMSC.USERID (특성 이름 상수) 및 XMSC.DELIVERY_AS_APP (값 상수) 입니다.

IBM.XMS.MQC struct를 사용하면 WebSphere MQ 상수에도 액세스할 수 있습니다. IBM.XMS 이름 공간이 이미 임포트된 경우, MQC.*<constant>*양식으로 이러한 등록 정보의 값에 액세스할 수 있습니다. 예를 들어, MQC.MQRO_COA_WITH_FULL_DATA입니다.

또한 .NET용 WebSphere MQ 클래스 및 XMS .NET를 사용하고 IBM.XMS 및 IBM.WMQ 네임스페이스를 가져오는 하이브리드 애플리케이션을 사용하는 경우, 각 발생이 고유하도록 MQC struct 네임스페이스에 완전한 자격을 부여해야 합니다.

관리되는 .NET 환경에서는 현재 일부 고급 기능이 지원되지 않습니다. 자세한 정보는 42 페이지의 『.NET의 관리되는 운영 및 관리되지 않는 운영』의 내용을 참조하십시오.

## .NET에 있는 존재하지 않는 특성 핸들링

XMS .NET에서 존재하지 않는 특성 처리는 대략적으로 JMS 스펙 및 XMS의 C 및 C++ 구현과 일치합니다.

JMS에서 존재하지 않는 특성에 액세스하면 메소드가 필수 유형으로 존재하지 않는(널) 값을 변환하려 할 때 Java 시스템 예외가 발생합니다. 특성이 존재하지 않는 경우 다음 예외가 발생합니다.

- getStringProperty 및 getObjectProperty에서 널을 리턴함
- Boolean.valueOf(null)가 false를 리턴하므로 getBooleanProperty가 false를 리턴함
- Integer.valueOf(null)가 예외를 처리하므로 getIntProperty 등이 java.lang.NumberFormatException을 처리함

XMS .NET에 특성이 존재하지 않는 경우 다음 예외가 발생합니다.

- GetStringProperty 및 GetObjectProperty(및 GetBytesProperty)에서 널을 리턴함(이는 Java와 동일함)
- GetBooleanProperty가 System.NullReferenceException을 처리함
- GetIntProperty 등이 System.NullReferenceException을 처리함

이 구현은 Java와 다르지만 대략적으로 JMS 스펙 및 XMS C/C++ 인터페이스와 일치합니다. Java 구현에서와 같이 XMS .NET는 System.Convert 호출의 예외를 호출자에게 전파합니다. 그러나 Java와는 달리 XMS는 .NET 프레임워크의 고유 작동을 사용하는 대신 시스템 변환 루틴에 널을 전달함으로써 NullReferenceExceptions를 명시적으로 전달합니다. 애플리케이션이 "abc"와 같은 문자열로 특성을 설정하고 GetIntProperty를 호출하면, Convert.ToInt32("abc")에서 전달된 System.FormatException이 호출자에게 전파되고 이는 Java와 일치합니다. MessageFormatException은 setProperty와 getProperty에 사용된 유형이 호환되지 않는 경우에만 처리됩니다. 이러한 작동은 Java와도 일치합니다.

## .NET의 오류 처리

XMS .NET 예외는 모두 System.Exception에서 파생됩니다. XMS 메소드 호출은 MessageFormatException과 같은 특정 XMS 예외, 일반 XMSException 또는 NullReferenceException과 같은 시스템 예외를 처리할 수 있습니다.

애플리케이션 요구사항에 적합한 대로 특정 캐치 블록에서 또는 일반 `System.Exception` 캐치 블록에서 이러한 오류를 캐치하는 애플리케이션을 작성하십시오.

## .NET의 메시지 및 예외 리스너

.NET 애플리케이션은 메시지 리스너를 사용하여 메시지를 비동기로 수신하고 예외 리스너를 사용하여 연결 문제점을 비동기로 알립니다.

메시지 및 예외 리스너의 기능은 .NET 및 C++에 동일합니다. 하지만 몇 가지 작은 구현의 차이가 있습니다.

## .NET의 메시지 리스너

메시지를 비동기적으로 수신하려면 다음 단계를 완료해야 합니다.

1. 메시지 리스너 위임의 서명과 일치하는 메소드를 정의하십시오. 정의할 메소드는 정적 또는 인스턴스 메소드가 될 수 있으며 액세스 가능한 클래스에 정의할 수 있습니다. 위임 서명은 다음과 같습니다.

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

   그러므로 메소드를 다음으로 정의할 수 있습니다.

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

2. 다음과 유사한 내용을 사용하여 이 메소드를 위임으로 인스턴스화하십시오.

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

3. 다음과 같이 위임을 이용자의 MessageListener 특성으로 설정하여 하나 이상의 이용자에 등록하십시오.

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

   MessageListener를 다시 널로 설정하여 위임을 제거할 수 있습니다.

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

## .NET의 예외 리스너

예외 리스너는 메시지 리스너와 상당히 유사한 방식으로 작동하지만 위임 정의가 다르며 메시지 이용자 이외의 연결에 지정됩니다. 이는 C++에서도 동일합니다.

1. 메소드를 정의하십시오. 위임 서명은 다음과 같습니다.

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

   그러므로 메소드를 다음으로 정의할 수 있습니다.

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

2. 다음과 유사하게 이 메소드를 위임으로 인스턴스화하십시오.

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

3. ExceptionListener 특성을 설정하여 위임을 연결에 등록하십시오.

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

   ExceptionListener를 다음과 같이 재설정하여 위임을 제거할 수 있습니다.

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

이에 대한 참조가 없는 경우, 시스템 가비지 콜렉터에서 예외 또는 메시지를 자동으로 삭제합니다.

다음은 샘플 코드입니다.

```
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);
    }
}
```

# 관리 대상 오브젝트에 대한 작업

이 장에서는 관리 오브젝트에 대한 정보를 제공합니다. XMS 애플리케이션은 중앙 관리 대상 오브젝트 저장소에서 오브젝트 정의를 검색하여 연결 팩토리와 목적지를 작성하는 데 사용할 수 있습니다.

장에는 다음 절가 포함됩니다.



**관련 개념**

관리 대상 오브젝트

관리 대상 오브젝트를 사용하여 중앙 저장소에서 관리 대상 클라이언트 애플리케이션이 사용하는 연결 설정을 관리할 수 있습니다. 애플리케이션은 중앙 저장소에서 오브젝트 정의를 검색하고 이를 사용하여

ConnectionFactory 및 Destination 오브젝트를 작성합니다. 관리 대상 오브젝트를 사용하면 런타임 시 사용하는 자원에서 애플리케이션을 커플링 해제할 수 있습니다.

## 관리 대상 오브젝트 저장소의 지원되는 유형

파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

파일 시스템 오브젝트 디렉토리는 직렬화된 JNDI(Java and Naming Directory Interface) 오브젝트 형식을 취합니다. LDAP 오브젝트 디렉토리는 JNDI 오브젝트를 포함하는 디렉토리입니다. 파일 시스템 및 LDAP 오브젝트 디렉토리는 WebSphere MQ v6.0과 제공되는 JMSAdmin 도구를 사용하여 또는 WebSphere MQ v7.0 이상과 제공되는 WebSphere MQ 탐색기를 사용하여 관리할 수 있습니다. 파일 시스템과 LDAP 오브젝트 디렉토리를 둘 다 WebSphere MQ 연결 팩토리 및 대상을 중앙 집중화하여 클라이언트 연결을 관리하는 데 사용할 수 있습니다. 네트워크 관리자는 동일한 중앙 저장소를 참조하고 중앙 저장소에서 작성한 연결 설정의 변경사항을 반영하도록 자동으로 업데이트되는 여러 애플리케이션을 배치할 수 있습니다.

COS 이름 지정 디렉토리에는 WebSphere 서비스 통합 버스 연결 팩토리 및 목적지가 포함되며 WebSphere 애플리케이션 서버 관리 콘솔을 사용하여 관리할 수 있습니다. XMS 애플리케이션에서 COS 네이밍 디렉토리에서 오브젝트를 검색하려면 JNDI 검색 웹 서비스를 배치해야 합니다. 이 웹 서비스는 일부 WebSphere 서비스 통합 기술에서 사용할 수 없습니다. 자세한 내용은 제품 문서를 참조하십시오.

**참고:** 오브젝트 디렉토리의 변경사항을 적용하려면 애플리케이션 연결을 다시 시작하십시오.

**관련 개념**

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

관리 대상 오브젝트
관리 대상 오브젝트를 사용하여 중앙 저장소에서 관리 대상 클라이언트 애플리케이션이 사용하는 연결 설정을 관리할 수 있습니다. 애플리케이션은 중앙 저장소에서 오브젝트 정의를 검색하고 이를 사용하여 ConnectionFactory 및 Destination 오브젝트를 작성합니다. 관리 대상 오브젝트를 사용하면 런타임 시 사용하는 자원에서 애플리케이션을 커플링 해제할 수 있습니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성

애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

## 관리 대상 오브젝트의 특성 맵핑

애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

예를 들어 WebSphere MQ JMS 연결 팩토리에서 검색된 특성이 있는 XMS 연결 팩토리를 작성하려면 이들 두 오브젝트 사이에 특성을 맵핑해야 합니다.

모든 특성 맵핑은 자동으로 수행됩니다.

다음은 연결 팩토리와 목적지의 가장 일반적인 일부 특성 간 맵핑을 보여 주는 표입니다. 이 표에서 표시되는 특성은 아주 적은 예일 뿐이며, 여기 표시되는 모든 특성이 연결 유형 및 서버 전체와 관련된 것도 아닙니다.

표 *8.* 연결 팩토리 및 목적지 특성의 이름 맵핑 예

| **WebSphere MQ JMS 특성 이름** | **XMS 특성 이름** | **WebSphere 서비스 통합 버스 특성 이름** |
|---|---|---|
| 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 |

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트

애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 관리 대상 ConnectionFactory 오브젝트의 필수 특성

애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

아래 표에 나열된 특성은 애플리케이션이 메시징 서버로의 연결을 작성하도록 설정하는 데 필요한 최소한의 특성입니다. 연결을 작성하는 방법을 사용자 정의하기 위해 애플리케이션이 ConnectionFactory 오브젝트의 추가 특성을 필요에 따라 설정할 수 있습니다. 추가 정보는 169 페이지의 『ConnectionFactory의 특성』의 내용을 참조하십시오. 사용 가능한 특성의 전체 목록이 포함됩니다.

### WebSphere MQ 큐 관리자에 연결

표 *9. WebSphere MQ* 큐 관리자에 대한 연결에 필요한 관리 *ConnectionFactory* 오브젝트의 특성 설정

| **필요한 XMS** | **필요한 동등한 WebSphere MQ JMS 특성** |
|---|---|
| XMSC_CONNECTION_TYPE | XMS가 연결 팩토리 클래스 이름 및 TRANSPORT(TRAN) 특성에서 이에 대해 작업합니다. |
| XMSC_WMQ_HOST_NAME | HOSTNAME(HOST) |
| XMSC_WMQ_PORT | 포트 |
| XMSC_WMQ_QUEUE_MANAGER | 큐 관리자의 이름 |

### 브로커에 대한 실시간 연결

표 *10.* 브로커에 대한 실시간 연결에 필요한 관리 대상 *ConnectionFactory* 오브젝트의 특성 설정

| **필요한 XMS** | **필요한 동등한 WebSphere MQ JMS 특성** |
|---|---|
| XMSC_CONNECTION_TYPE | XMS가 연결 팩토리 클래스 이름 및 TRANSPORT(TRAN) 특성에서 이에 대해 작업합니다. |
| XMSC_RTT_HOST_NAME | HOSTNAME(HOST) |
| XMSC_RTT_PORT | 포트 |

## WebSphere 서비스 통합 버스에 대한 연결

| 표 *11. WebSphere* 서비스 통합 버스에 대한 연결에 필요한 관리된 *ConnectionFactory* 오브젝트의 특성 설정 | |
|---|---|
| **XMS 특성** | **설명** |
| XMSC_CONNECTION_TYPE | 애플리케이션이 연결되는 메시징 서버의 유형입니다. . 이는 연결 팩토리 클래스 이름에서 결정됩니다. |
| XMSC_WPM_BUS_NAME | 연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다. |

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

WebSphere MQ 큐 관리자에 대한 보안 연결
XMS .NET 애플리케이션이 WebSphere MQ 큐 관리자에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

WebSphere 서비스 통합 버스 메시징 엔진에 대한 보안 연결
XMS 애플리케이션이 WebSphere 서비스 통합 버스 메시징 엔진에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 관리 대상 Destination 오브젝트의 필수 특성

목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

표 *12.* 관리 대상 *Destination* 오브젝트의 특성 설정

| 연결 유형 | 특성 | 설명 |
|---|---|---|
| WebSphere MQ 큐 관리자 | QUEUE(QU)<br>TOPIC(TOP) | 연결하려는 큐<br>애플리케이션이 목적지로 사용하는 토픽 |
| 브로커에 대한 실시간 연결 | TOPIC(TOP) | 애플리케이션이 목적지로 사용하는 토픽 |
| WebSphere 서비스 통합 버스 | topicName<br>queueName | 애플리케이션이 토픽에 연결되는 경우<br>애플리케이션이 큐에 연결되는 경우 |

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

IDestination(.NET 인터페이스의 경우)

목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## 관리 대상 오브젝트 작성

XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

### 시작하기 전에

XMS가 지원하는 여러 유형의 관리 대상 오브젝트 저장소에 대한 자세한 정보는 47 페이지의 『관리 대상 오브젝트 저장소의 지원되는 유형』의 내용을 참조하십시오.

### 이 태스크 정보

WebSphere MQ의 관리 오브젝트를 작성하려면, WebSphere MQ 탐색기 또는 WebSphere MQ JMS 관리 (JMSAdmin) 도구를 사용하십시오.

WebSphere MQ 또는 IBM Integration Bus의 관리 대상 오브젝트를 작성하려면 WebSphere MQ JMS 관리 (JMSAdmin) 도구를 사용하십시오.

WebSphere 서비스 통합 버스에 대한 관리 대상 오브젝트를 작성하려면 WebSphere 애플리케이션 서버 관리 콘솔을 사용하십시오.

아래 단계는 관리 대상 오브젝트 작성을 위해 수행하는 작업이 요약되어 있습니다.

### 프로시저

1. 연결 팩토리를 작성하고 애플리케이션에서 선택한 서버로의 연결을 작성하는 데 필요한 특성을 정의하십시오.

   XMS가 연결을 작성하는 데 필요한 최소 특성은 49 페이지의 『관리 대상 ConnectionFactory 오브젝트의 필수 특성』에 정의되어 있습니다.
2. 애플리케이션이 연결하는 메시징 서버에서 필수 목적지를 작성하십시오.
   - WebSphere MQ 큐 관리자에 연결하려면 큐 또는 토픽을 작성합니다.
   - 브로커에 대한 실시간 연결의 경우 토픽을 작성하십시오.
   - WebSphere 서비스 통합 버스에 연결하려면 큐 또는 토픽을 작성합니다.

   XMS가 연결을 작성하는 데 필요한 최소 특성은 51 페이지의 『관리 대상 Destination 오브젝트의 필수 특성』에 정의되어 있습니다.

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스

XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

관리 대상 오브젝트
관리 대상 오브젝트를 사용하여 중앙 저장소에서 관리 대상 클라이언트 애플리케이션이 사용하는 연결 설정을 관리할 수 있습니다. 애플리케이션은 중앙 저장소에서 오브젝트 정의를 검색하고 이를 사용하여 `ConnectionFactory` 및 `Destination` 오브젝트를 작성합니다. 관리 대상 오브젝트를 사용하면 런타임 시 사용하는 자원에서 애플리케이션을 커플링 해제할 수 있습니다.

ConnectionFactories 오브젝트 및 Connection 오브젝트
ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

**관련 태스크**

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

IDestination(.NET 인터페이스의 경우)
목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

Destination 특성
Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## InitialContext 오브젝트

애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

### 이 태스크 정보

InitialContext 오브젝트는 저장소에 대한 연결을 캡슐화합니다. XMS API는 다음 태스크를 수행하는 메소드를 제공합니다.

- InitialContext 오브젝트 작성
- 관리 대상 오브젝트 저장소에서 관리 대상 오브젝트 검색

InitialContext 오브젝트 작성에 대한 자세한 정보를 보려면 .NET용 101 페이지의 『InitialContext』 및 176 페이지의 『InitialContext 특성』의 내용을 참조하십시오.

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형

파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**
관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**
관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

InitialContext(.NET 인터페이스의 경우)
애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

InitialContext 특성
InitialContext 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## InitialContext 특성

InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS의 .NET 구현 및 JNDI에서, 추가 정보가 환경 해시 테이블로 구성자에게 제공됩니다.

관리 대상 오브젝트 저장소의 위치는 XMSC_IC_URL 특성에 정의되어 있습니다. 이 특성은 일반적으로 Create 호출에서 전달되지만 검색 전에 다른 이름 지정 디렉토리에 연결하기 위해 수정될 수도 있습니다. FileSystem 또는 LDAP 컨텍스트에서는 이 특성이 디렉토리의 주소를 정의합니다. COS 네이밍의 경우, 이 특성이 JNDI 디렉토리에 연결하는 데 이 특성을 사용하는 웹 서비스의 주소입니다.

다음 특성은 JNDI 디렉토리에 연결하는 데 사용할 웹 서비스에 수정되지 않은 상태로 전달됩니다.

- XMSC_IC_PROVIDER_URL
- XMSC_IC_SECURITY_CREDENTIALS
- XMSC_IC_SECURITY_AUTHENTICATION

- XMSC_IC_SECURITY_PRINCIPAL
- XMSC_IC_SECURITY_PROTOCOL

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

InitialContext(.NET 인터페이스의 경우)
애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

InitialContext 특성
InitialContext 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## XMS 초기 컨텍스트의 URI 형식

관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

### FileSystem 컨텍스트

FileSystem 컨텍스트의 경우 URL은 파일 시스템 기반 디렉토리의 위치를 제공합니다. URL의 구조는 RFC 1738, *URL(Uniform Resource Locator)*로 정의됩니다. URL에는 `file://` 접두부가 있으며 이 접두부 다음에 나오는 구문은 XMS가 실행 중인 시스템에서 열릴 수 있는 유효한 파일 정의입니다.

이 구문은 플랫폼에 따라 달라질 수 있으며 '/' 구분 기호 또는 '\' 구분 기호를 사용할 수 있습니다. '\'를 사용하는 경우 각 구분 기호는 '\'를 하나 더 사용하여 이스케이프되어야 합니다. 그러면 .NET Framework가 해당 구분 기호를 뒤에 오는 내용을 위한 이스케이프 문자로 해석하는 것을 방지합니다.

이 예는 다음 구문을 설명합니다.

```
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 컨텍스트

LDAP 컨텍스트의 경우 URL의 기본 구조는 대소문자를 구분하지 않는 접두부 `ldap://`를 사용하는 RFC 2255, *LDAP URL* 형식으로 정의됩니다.

다음 예제에는 정확한 구문이 설명되어 있습니다.

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

이 구문은 RFC로 정의되어 있으나 구문에 속성, 범위, 필터 또는 확장자에 대한 지원이 없습니다.

이 구문의 예제는 다음과 같습니다.

```
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 컨텍스트

WSS 컨텍스트의 경우, URL은 `http://` 접두부를 사용하는 웹 서비스 엔드포인트의 형식입니다.

또는 `cosnaming://`이나 `wsvc://` 접두부를 사용할 수도 있습니다.

이 두 접두부는 http를 통해 액세스되는 URL이 포함된 WSS 컨텍스트가 사용 중인 것으로 해석됩니다. 이를 통해 URL에서 직접 초기 컨텍스트 유형을 쉽게 파생시킬 수 있습니다.

이 구문의 예를 들면 다음과 같습니다.

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

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

관리 대상 오브젝트의 검색

XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

InitialContext(.NET 인터페이스의 경우)
애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

InitialContext 특성
InitialContext 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

## JNDI 검색 웹 서비스

XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

이 웹 서비스는 설치 디렉토리에 있는 엔터프라이즈 아카이브 파일 SIBXJndiLookupEAR.ear에서 제공됩니다. Message Service Client for .NET의 현재 릴리스의 경우, SIBXJndiLookupEAR.ear을 `<install_dir>\java\lib` 디렉토리에서 찾을 수 있습니다. 관리 콘솔이나 wsaadmin 스크립팅 도구를 사용하여 WebSphere 서버에서 설치할 수 있습니다. 웹 서비스 애플리케이션 배치에 관한 자세한 정보는 제품 문서를 참조하십시오.

XMS 애플리케이션 내에서 웹 서비스를 정의하려면 InitialContext 오브젝트의 XMSC_IC_URL 특성을 웹 서비스 엔드포인트 URL로 설정해야 합니다. 예를 들어, 웹 서비스를 MyServer라는 서버 호스트에 배치하는 경우, 웹 서비스 엔드포인트 URL의 예는 다음과 같습니다.

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

XMSC_IC_URL 특성을 설정하면 InitialContext Lookup 호출이 정의된 엔드포인트에서 웹 서비스를 호출할 수 있으며, 순차적으로 COS 네이밍 서비스에서 필수 관리 오브젝트를 검색합니다.

.NET 애플리케이션은 웹 서비스를 사용할 수 있습니다. 서버 측 배치는 XMS C, /C++ 및 XMS .NET에서 동일합니다.XMS .NET 는 마이크로소프트 .NET 프레임워크를 통해 웹 서비스를 직접 호출합니다.

**관련 개념**

관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성

InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

메시징 서버 환경 설정
이 장에서는 XMS 애플리케이션이 서버에 연결할 수 있도록 메시징 서버 환경을 설정하는 방법에 대해 설명합니다.

XMS 샘플 애플리케이션 사용
XMS와 함께 제공된 샘플 애플리케이션을 사용하여 설치 및 메시징 서버 설정을 확인하고 고유한 애플리케이션을 빌드하도록 도와줍니다. 샘플은 각 API의 공통 기능에 대해 간략하게 설명합니다.

**관련 태스크**
관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

설치 마법사를 사용하여 Message Service Client for .NET 설치
설치에 InstallShield X/Windows MSI 설치 프로그램을 사용합니다. 전체 설치 또는 사용자 정의 설치의 두 가지 설치 옵션을 사용할 수 있습니다.

**관련 참조**
관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

## 관리 대상 오브젝트의 검색

XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

검색될 오브젝트의 이름 유형은 다음과 같습니다.

- Destination 오브젝트를 설명하는 단순한 이름. 큐 목적지 이름인 SalesOrders를 예로 들 수 있습니다.
- SubContext로 구성될 수 있는 컴포지트 이름. '/'로 구분되며 오브젝트 이름으로 끝나야 합니다. "Warehouse/PickLists/DispatchQueue2" 컴포지트 이름을 예로 들 수 있습니다. 여기서 Warehouse와 Picklists는 이름 지정 디렉토리의 서브컨텍스트이며 DispatchQueue2는 Destination 오브젝트의 이름입니다.

**관련 개념**
관리 대상 오브젝트 저장소의 지원되는 유형
파일 시스템 및 LDAP 관리 대상 오브젝트는 WebSphere MQ 및 WebSphere 애플리케이션 서버에 연결할 때 사용될 수 있는 반면, COS 이름 지정은 WebSphere 애플리케이션 서버에 연결할 때에만 사용될 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

InitialContext 특성

InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

JNDI 검색 웹 서비스
XMS에서 COS 네이밍 디렉토리에 액세스하려면 JNDI 검색 웹 서비스를 WebSphere 서비스 통합 버스 서버에 배치해야 합니다. 이 웹 서비스는 COS 네이밍 서비스의 Java 정보를 XMS 애플리케이션이 읽을 수 있는 형식으로 변환합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

InitialContext(.NET 인터페이스의 경우)
애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

InitialContext 특성
InitialContext 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

# XMS 애플리케이션의 통신에 보안 설정

이 장에서는 XMS 애플리케이션이 SSL(Secure Sockets Layer)를 통해 WebSphere 서비스 통합 버스 메시징 엔징 또는 WebSphere MQ 큐 관리자에 연결할 수 있도록 보안 통신을 설정하는 방법에 대한 정보를 제공합니다.

장에는 다음 절가 포함됩니다.


- 59 페이지의 『WebSphere MQ 큐 관리자에 대한 보안 연결』
- 60 페이지의 『WebSphere MQ 큐 관리자에 대한 연결에 필요한 CipherSuite 및 CipherSpec 이름 맵핑』
- 62 페이지의 『WebSphere 서비스 통합 버스 메시징 엔진에 대한 보안 연결』
- 62 페이지의 『WebSphere 서비스 통합 버스에 대한 연결에 필요한 CipherSuite 및 CipherSpec 이름 맵핑』


## WebSphere MQ 큐 관리자에 대한 보안 연결

XMS .NET 애플리케이션이 WebSphere MQ 큐 관리자에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

암호화 조정에 사용되는 프로토콜은 ConnectionFactory 오브젝트에 지정되는 CipherSuite에 따라 SSL(Secure Sockets Layer) 또는 TLS(Transport Layer Security)가 될 수도 있습니다.

WebSphere MQ 버전 7.0.0.1 이상 클라이언트 라이브러리를 사용하고 WebSphere MQ 버전 7 큐 관리자에 연결하는 경우, XMS 애플리케이션에서 동일한 큐 관리자에 대해 다중 연결을 작성할 수 있습니다. 그러나 다른 큐 관리자에 대한 연결은 허용되지 않습니다. 이를 시도하면 `MQRC_SSL_ALREADY_INITIALIZED` 오류가 발생합니다.

e WebSphere MQ 버전 6 이상 클라이언트 라이브러리를 사용하면, 이전 SSL 연결을 먼저 닫는 경우에만 SSL 연결을 작성할 수 있습니다. 동일한 프로세스에서 동일한 또는 다른 큐 관리자에 다중 SSL 연결을 동시에 작성할 수 없습니다. 요청을 두 개 이상 시도하는 경우 요청된 일부 SSL 연결 매개변수가 무시되었다는 내용의 MQRC_SSL_ALREADY_INITIALIZED 경고가 발생합니다.

SSL를 통해 IBM WebSphere MQ 관리자에 대한 연결의 ConnectionFactory 특성이 간단한 설명과 함께 다음 표에 표시됩니다.

표 *13. SSL*을 통한 *WebSphere MQ* 큐 관리자에 대한 연결의 *ConnectionFactory* 특성

| 특성 이름 | 설명 |
|---|---|
| XMSC_WMQ_SSL_CERT_STORES | 큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 인증서 폐기 목록(CRL)이 있는 서버의 위치입니다. |
| XMSC_WMQ_SSL_CIPHER_SPEC | 큐 관리자에 대한 보안 연결에 사용할 CipherSpec의 이름. |
| XMSC_WMQ_SSL_CIPHER_SUITE | 큐 관리자에 대한 SSL 또는 TLS 연결에 사용될 CipherSuite 의 이름입니다. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다. |
| XMSC_WMQ_SSL_CRYPTO_HW | 클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항입니다. |
| XMSC_WMQ_SSL_FIPS_REQUIRED | 이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 또는 사용할 수 없는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 FIPS 알고리즘만 클라이언트-서버 연결에 사용됩니다. |
| XMSC_WMQ_SSL_KEY_REPOSITORY | 키 및 인증서가 저장되는 키 데이터베이스 파일의 위치입니다. |
| XMSC_WMQ_SSL_KEY_RESETCOUNT | KeyResetCount는 비밀 키가 재협상되기 전에 SSL 통신에서 송신되고 수신된 암호화되지 않은 총 바이트 수를 나타냅니다. |
| XMSC_WMQ_SSL_PEER_NAME | 큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 피어 이름입니다. |

**관련 참조**

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

## *WebSphere MQ* 큐 관리자에 대한 연결에 필요한 *CipherSuite* 및 *CipherSpec* 이름 맵핑

InitialContext는 JMSAdmin Connection Factory 특성 SSLCIPHERSUITE와 XMS와 거의 동일한 XMSC_WMQ_SSL_CIPHER_SPEC 사이에서 변환됩니다. XMSC_WMQ_SSL_CIPHER_SUITE 값은 지정하지만 XMSC_WMQ_SSL_CIPHER_SPEC 값은 생략하는 경우 유사한 변환이 필요합니다.

60 페이지의 표 14에서는 사용 가능한 CipherSpecs 및 해당하는 동등한 JSSE CipherSuite를 나열합니다.

표 *14.* 사용 가능한 *CipherSpec* 및 해당하는 동등한 *JSSE CipherSuite*

| CipherSpec | 동등한 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 |

| 표 *14.* 사용 가능한 *CipherSpec* 및 해당하는 동등한 *JSSE CipherSuite (*계속*)* | |
|---|---|
| **CipherSpec** | **동등한 JSSE CipherSuite** |
| 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 |
| 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 |

**참고:** CipherSuite 이름 SSL_RSA_WITH_3DES_EDE_CBC_SHA 또는 SSL_RSA_WITH_DES_CBC_SHA에 대한 일대일 맵핑은 특성 XMSC_WMQ_SSL_FIPSREQUIRED의 설정을 설명하고 휴리스틱을 적용해야 합니다.

>특성 XMSC_WMQ_SSL_CIPHER_SUITE에 대해 SSL_RSA_WITH_3DES_EDE_CBC_SHA 또는 SSL_RSA_WITH_DES_CBC_SHA를 지정하고 XMSC_WMQ_SSL_CIPHER_SPEC에 대해 값이 없는 경우, XMSC_WMQ_SSL_CIPHER_SPEC에 대한 값이 다음 테이블에 따라 선택됩니다.

XMSC_WMQ_SSL_CIPHER_SUITE 특성에 대해 SSL_RSA_WITH_3DES_EDE_CBC_SHA를 지정할 때 XMSC_WMQ_SSL_CIPHER_SPEC에 대해 사용된 값은 다음 테이블에 표시됩니다.

| 표 *15. XMSC_WMQ_SSL_CIPHER_SUITE* 특성에 *SSL_RSA_WITH_3DES_EDE_CBC_SHA*를 지정하는 경우 *XMSC_WMQ_SSL_CIPHER_SPEC*에 사용되는 값 | |
|---|---|
| **입력: XMSC_WMQ_SSL_FIPSREQUIRED 값** | **출력: 선택된 XMSC_WMQ_SSL_CIPHER_SPEC** |
| false(즉, MQSSL_FIPS_NO) | TRIPLE_DES_SHA_US |
| true(즉, MQSSL_FIPS_YES) | TLS_RSA_WITH_3DES_EDE_CBC_SHA |

**참고:**

- TLS_RSA_WITH_3DES_EDE_CBC_SHA는 더 이상 사용되지 않습니다. 그러나 AMQ9288 오류로 인해 연결이 종료되기 전까지는 이 CipherSpec을 사용하여 최대 32GB의 데이터를 전송할 수 있습니다. 이 오류를 방지하려면 3중 DES를 사용하지 않거나 이 CipherSpec을 사용할 때 비밀 키 재설정을 사용 가능하게 하십시오.

XMSC_WMQ_SSL_CIPHER_SUITE 특성에 대해 SSL_RSA_WITH_DES_CBC_SHA를 지정할 때 XMSC_WMQ_SSL_CIPHER_SPEC에 대해 사용된 값은 다음 테이블에 표시됩니다.

| 표 *16. XMSC_WMQ_SSL_CIPHER_SUITE* 특성에 *SSL_RSA_WITH_DES_CBC_SHA*를 지정하는 경우 *XMSC_WMQ_SSL_CIPHER_SPEC*에 사용되는 값 | |
|---|---|
| **입력: XMSC_WMQ_SSL_FIPSREQUIRED 값** | **출력: 선택된 XMSC_WMQ_SSL_CIPHER_SPEC** |
| false(즉, MQSSL_FIPS_NO) | DES_SHA_EXPORT |
| true(즉, MQSSL_FIPS_YES) | TLS_RSA_WITH_DES_CBC_SHA |

## WebSphere 서비스 통합 버스 메시징 엔진에 대한 보안 연결

XMS 애플리케이션이 WebSphere 서비스 통합 버스 메시징 엔진에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

XMS는 WebSphere 서비스 통합 버스에 연결하는 데 필요한 SSL 및 HTTPS 지원을 제공합니다. SSL 및 HTTPS는 인증 및 기밀성을 위해 보안 연결을 제공합니다.

WebSphere 보안과 유사하게 XMS 보안은 JSSE 보안 표준 및 이름 지정 규칙을 준수하도록 구성되며, 여기에는 보안 연결을 조정할 때 사용되는 알고리즘을 지정하는 CipherSuite 사용이 포함됩니다. 암호화 조정에 사용되는 프로토콜은 ConnectionFactory 오브젝트에 지정되는 CipherSuite에 따라 SSL 또는 TLS가 될 수 있습니다.

62 페이지의 표 17에는 ConnectionFactory 오브젝트에 정의되어야 하는 특성이 나열되어 있습니다.

표 *17. WebSphere* 서비스 통합 버스 메시징 엔진에 보안 연결하기 위한 *ConnectionFactory*의 특성

| 특성 이름 | 설명 |
|---|---|
| XMSC_WPM_SSL_CIPHER_SUITE | WebSphere 서비스 통합 버스 메시징 엔진에 대한 SSL 또는 TLS 연결에 사용할 CipherSuite의 이름. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다. |
| XMSC_WPM_SSL_KEYRING_LABEL | 서버 인증 시 사용되는 인증서입니다. |

다음은 WebSphere 통합 메시징 엔진에 보안 연결하기 위한 ConnectionFactory 특성의 예입니다.

```
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);
```

여기서 chain_name은 BootstrapTunneledSecureMessaging 또는 BootstrapSecureMessaging 중 하나로 설정되어야 하고, port_number는 부트스트랩 서버가 수신 요청을 청취하는 포트 번호입니다.

다음은 삽입된 샘플 값을 사용하여 WebSphere 통합 메시징 엔진에 보안 연결하기 위한 ConnectionFactory 특성의 예입니다.

```
  /* 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");
```

**관련 참조**

IConnectionFactory(.NET 인터페이스의 경우)
애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

ConnectionFactory의 특성
ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

## *WebSphere* 서비스 통합 버스에 대한 연결에 필요한 *CipherSuite* 및 *CipherSpec* 이름 맵핑

GSKit은 CipherSuite가 아니라 CipherSpec을 사용하므로 XMSC_WPM_SSL_CIPHER_SUITE 특성에 지정된 JSSE 스타일의 CipherSuite 이름을 GSKit 스타일의 CipherSpec 이름으로 맵핑해야 합니다.

63 페이지의 표 18에서는 인식된 각 CipherSuite에 대해 동등한 CipherSpec을 나열합니다.

표 18. 사용 가능한 *CipherSuite* 및 해당하는 동등한 *CipherSpec*

| **CipherSuite** | **동등한 CipherSpec** |
|---|---|
| SSL_RSA_WITH_NULL_MD5 | NULL_MD5 |
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | RC4_MD5_EXPORT |
| SSL_RSA_WITH_RC4_128_MD5 | RC4_MD5_US |
| SSL_RSA_WITH_NULL_SHA | NULL_SHA |
| SSL_RSA_EXPORT1024_WITH_RC4_56_SHA | RC4_56_SHA_EXPORT1024 |
| SSL_RSA_WITH_RC4_128_SHA | RC4_SHA_US |
| SSL_RSA_WITH_DES_CBC_SHA | DES_SHA_EXPORT |
| SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA | DES_SHA_EXPORT1024 |
| SSL_RSA_FIPS_WITH_DES_CBC_SHA | FIPS_WITH_DES_CBC_SHA |
| SSL_RSA_WITH_3DES_EDE_CBC_SHA | TRIPLE_DES_SHA_US |
| SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA | FIPS_WITH_3DES_EDE_CBC_SHA |
| TLS_RSA_WITH_DES_CBC_SHA | TLS_RSA_WITH_DES_CBC_SHA |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA | TLS_RSA_WITH_3DES_EDE_CBC_SHA |
| TLS_RSA_WITH_AES_128_CBC_SHA | TLS_RSA_WITH_AES_128_CBC_SHA |
| TLS_RSA_WITH_AES_256_CBC_SHA | TLS_RSA_WITH_AES_256_CBC_SHA |

**참고:**

- TLS_RSA_WITH_3DES_EDE_CBC_SHA는 더 이상 사용되지 않습니다. 그러나 AMQ9288 오류로 인해 연결이 종료되기 전까지는 이 CipherSpec을 사용하여 최대 32GB의 데이터를 전송할 수 있습니다. 이 오류를 방지하려면 3중 DES를 사용하지 않거나 이 CipherSpec을 사용할 때 비밀 키 재설정을 사용 가능하게 하십시오.

# XMS 메시지

이 장에서는 XMS 메시지의 구조와 컨텐츠에 대해 설명하고 애플리케이션에서 XMS 메시지를 처리하는 방법을 제공합니다.

이 장에는 다음 절가 포함됩니다.



**관련 참조**

IMessage(.NET 인터페이스의 경우)
Message 오브젝트는 애플리케이션이 송신하거나 수신하는 메시지를 나타냅니다. IMessage는 IMapMessage와 같은 메시지 클래스의 수퍼클래스입니다.

## XMS 메시지의 파트

XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

**헤더**
메시지의 헤더에는 필드가 있으며 모든 메시지에는 동일한 세트의 헤더 필드가 있습니다. XMS 및 애플리케이션은 헤더 필드 값을 사용하여 메시지를 식별하고 라우팅합니다. 헤더 필드에 대한 자세한 정보는 64 페이지의 『XMS 메시지의 헤더 필드』의 내용을 참조하십시오.

**특성 세트**
메시지 특성은 메시지에 대한 추가 정보를 지정합니다. 모든 메시지에 동일한 헤더 필드 세트가 있긴 하지만 모든 메시지의 특성 세트는 서로 다를 수 있습니다. 추가 정보는 65 페이지의 『XMS 메시지의 특성』의 내용을 참조하십시오.

**Body**
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 추가 정보는 68 페이지의 『XMS 메시지 본문』의 내용을 참조하십시오.

애플리케이션에서 수신할 메시지를 선택할 수 있습니다. 선택 기준을 지정하는 메시지 선택자를 사용합니다. 이 기준은 특정 헤더 필드 값 및 메시지의 특성 값에 기반할 수 있습니다. 메시지 선택자에 대한 자세한 정보는 73 페이지의 『메시지 선택자』의 내용을 참조하십시오.

**관련 참조**
XMS 메시지의 헤더 필드
XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

XMS 메시지의 특성
XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

XMS 메시지 본문
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

메시지 선택자
XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

XMS 메시지를 WebSphere MQ 메시지로 맵핑
XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

## XMS 메시지의 헤더 필드

XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

이 헤더 필드의 이름은 JMS 접두부로 시작합니다. JMS 메시지 헤드 필드에 대한 설명을 보려면 *Java Message Service Specification,* 버전 *1.1*의 내용을 참조하십시오.

XMS는 implements the JMS 메시지 헤더 필드를 Message 오브젝트의 속성으로 구현합니다. 각 헤더 필드에는 해당 값을 설정하고 가져오는 데 필요한 자체 메소드가 있습니다. 이 메소드에 대한 설명을 보려면 113 페이지의 『IMessage』의 내용을 참조하십시오. 헤더 필드는 항상 읽기/쓰기가 가능합니다.

64 페이지의 표 19에는 JMS 메시지 헤더 필드가 나열되고 전송된 메시지에 대해 각 필드의 값을 설정하는 방법을 표시됩니다. 필드 중 일부는 애플리케이션이 메시지를 전송할 때 또는 JMSRedelivered의 경우 애플리케이션이 메시지를 수신할 때 XMS에 의해 자동으로 설정됩니다.

표 *19. JMS* 메시지 헤더 필드

| **JMS 메시지 헤더 필드의 이름** | **전송된 메시지에 값을 설정하는 방법(*method* [*class*] 형식)** |
|---|---|
| JMSCorrelationID | Set JMSCorrelationID [Message] |
| JMSDeliveryMode | Send [MessageProducer] |
| JMSDestination | Send [MessageProducer] |
| JMSExpiration | Send [MessageProducer] |
| JMSMessageID | Send [MessageProducer] |

| 표 *19. JMS* 메시지 헤더 필드 *(계속)* | |
|---|---|
| **JMS 메시지 헤더 필드의 이름** | **전송된 메시지에 값을 설정하는 방법(*method* [*class*] 형식)** |
| JMSPriority | Send [MessageProducer] |
| JMSRedelivered | Receive [MessageConsumer] |
| JMSReplyTo | Set JMSReplyTo [Message] |
| JMSTimestamp | Send [MessageProducer] |
| JMSType | Set JMSType [Message] |

**관련 참조**

XMS 메시지의 파트
XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

XMS 메시지의 특성
XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

XMS 메시지 본문
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

메시지 선택자
XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

XMS 메시지를 WebSphere MQ 메시지로 맵핑
XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

## XMS 메시지의 특성

XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

XMS에서 다음과 같은 메시지 오브젝트의 사전 정의된 특성을 지원하므로 XMS 애플리케이션은 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있습니다.

- WebSphere JMS에서 지원하는 동일한 JMS 정의 특성. 이러한 특성의 이름은 JMSX 접두부로 시작합니다.
- WebSphere JMS에서 지원하는 동일한 IBM 정의 특성. 이러한 특성의 이름은 JMS_IBM_ 접두부로 시작합니다.

각 사전 정의된 특성의 이름은 다음과 같이 두 개입니다.

- JMS 이름(JMS 정의 특성 또는 WebSphere JMS 이름(IBM 정의 특성)

  이는 특성이 JMS 또는 WebSphere JMS에서 알려진 이름이며, 또한 이 특성이 있는 메시지와 함께 전송되는 이름입니다. XMS 애플리케이션은 이 이름을 사용하여 메시지 선택자 표현식의 특성을 식별합니다.
- 메시지 선택자 표현식을 제외한 모든 상황에서 특성을 식별하기 위한 XMS 이름. 각 XMS 이름은 `IBM.XMS.XMSC` 클래스에서 이름 지정된 상수로서 정의됩니다. 이름 지정된 상수의 값은 해당하는 JMS 또는 WebSphere JMS 이름입니다.

사전 정의된 특성에 추가로, XMS 애플리케이션은 고유한 메시지 특성 세트를 작성하고 사용할 수 있습니다. 이러한 특성을 애플리케이션 정의 특성이라고 합니다.

애플리케이션이 메시지를 작성한 후에 메시지의 특성을 읽고 쓸 수 있습니다. 애플리케이션이 메시지를 전송한 후에도 특성은 계속 읽고 쓸 수 있습니다. 애플리케이션이 메시지를 수신할 경우 메시지 특성은 읽기 전용입니다. 메시지 특성이 읽기 전용일 때 애플리케이션이 메시지 클래스의 `Clear Properties` 메소드를 호출하는 경우, 특성은 읽기 및 쓰기 가능하게 됩니다. 또한 이 메소드가 특성도 지웁니다.

메시지 특성을 제거한 후 전달될 때 수신되는 메시지는 메시지 특성이 제거된 표준 .NET용 WMQ XMS BytesMessage를 전달하는 것과 동일한 방식으로 작동합니다.

그러나 이는 다음 특성이 유실되므로 권장하지 않습니다.

- JMS_IBM_Encoding 특성 값 - 메시지 데이터가 의미 있게 디코딩될 수 없음을 의미합니다.

- JMS_IBM_Format 특성 값 - (MQMD 또는 새 MQRFH2) 메시지 헤더와 기존 헤더 사이의 헤더 체인이 끊어짐을 의미합니다.

메시지의 모든 특성 값을 판별하기 위해 애플리케이션이 Message 클래스의 Get Properties 메소드를 호출할 수 있습니다. 이 메소드는 Property 오브젝트의 목록을 캡슐화하는 반복기를 작성합니다. 각 Property 오브젝트는 메시지의 특성을 나타냅니다. 그런 다음 애플리케이션은 Iterator 클래스 메소드를 사용하여 각 Property 오브젝트를 차례로 검색할 수 있고 Property 클래스 메소드를 사용하여 각 특성의 이름, 데이터 유형 및 값을 검색할 수 있습니다.

**관련 참조**

XMS 메시지의 파트
XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

XMS 메시지의 헤더 필드
XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

XMS 메시지 본문
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

메시지 선택자
XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

XMS 메시지를 WebSphere MQ 메시지로 맵핑
XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

## 메시지의 *JMS* 정의 특성

일부 JMS 정의 메시지 특성이 XMS 및 WebSphere JMS에서 지원됩니다.

66 페이지의 표 20에는 XMS 및 WebSphere JMS에서 지원하는 JMS 정의 메시지 특성이 표시됩니다. JMS 정의 특성에 대한 설명은 *Java Message Service Specification,* 버전 *1.1*의 내용을 참조하십시오. JMS 정의 특성은 브로커에 대한 실시간 연결에는 유효하지 않습니다.

이 표는 각 특성의 데이터 유형을 지정하고 전송되는 메시지에 대해 특성 값을 설정하는 방법을 나타냅니다. 특성 중 일부는 애플리케이션이 메시지를 전송할 때 또는 JMSXDeliveryCount의 경우 애플리케이션이 메시지를 수신할 때 XMS에 의해 자동으로 설정됩니다.

표 *20.* 메시지의 *JMS* 정의 특성

| **JMS 정의 특성의 XMS 이름** | **JMS 이름** | **데이터 유형** | **전송된 메시지에 대한 값 설정 방법(*method* [*class*] 형식)** |
|---|---|---|---|
| JMSX_APPID | JMSXAppID | System.String | Send [MessageProducer] |
| JMSX_DELIVERY_COUNT | JMSXDeliveryCount | System.Int32 | Receive [MessageConsumer] |
| JMSX_GROUPID | JMSXGroupID | System.String | Set String Property [PropertyContext] |
| JMSX_GROUPSEQ | JMSXGroupSeq | System.Int32 | Set Integer Property [PropertyContext] |
| JMSX_USERID | JMSXUserID | System.String | Send [MessageProducer] |

## *IBM* 정의 메시지 특성

일부 IBM 정의 메시지 특성이 XMS 및 WebSphere JMS에서 지원됩니다.

67 페이지의 표 21에는 XMS 및 WebSphere JMS에서 지원하는 메시지의 IBM 정의 특성이 표시됩니다. IBM 정의 특성에 대한 자세한 정보는 *Java*를 사용하는 *IBM WebSphere MQ* 또는 WebSphere Application Server 제품 문서를 참조하십시오.

이 표는 각 특성의 데이터 유형을 지정하고 전송되는 메시지에 대해 특성 값을 설정하는 방법을 나타냅니다. 특성 중 일부는 애플리케이션이 메시지를 전송할 때 XMS에 의해 자동으로 설정됩니다.

표 *21. IBM* 정의 메시지 특성

| **IBM 정의 특성의 XMS 이름** | **WebSphere JMS 이름** | **데이터 유형** | **전송된 메시지에 대한 값 설정 방법(*method* [*class*] 형식)** |
|---|---|---|---|
| JMS_IBM_CHARACTER_SET | JMS_IBM_Character_Set | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_ENCODING | JMS_IBM_Encoding | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_EXCEPTIONMESSAGE | JMS_IBM_ExceptionMessage | System.String | Receive [MessageConsumer] |
| JMS_IBM_EXCEPTIONREASON | JMS_IBM_ExceptionReason | System.Int32 | Receive [MessageConsumer] |
| JMS_IBM_EXCEPTIONTIMESTAMP | JMS_IBM_ExceptionTimestamp | System.Int64 | Receive [MessageConsumer] |
| JMS_IBM_EXCEPTIONPROBLEM DESTINATION | JMS_IBM_ExceptionProblemDestination | System.String | Receive [MessageConsumer] |
| JMS_IBM_FEEDBACK | JMS_IBM_Feedback | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_FORMAT | JMS_IBM_Format | System.String | Set String Property [PropertyContext] |
| JMS_IBM_LAST_MSG_IN_GROUP | JMS_IBM_Last_Msg_In_Group | System.Boolean | Set Integer Property [PropertyContext] |
| JMS_IBM_MSGTYPE | JMS_IBM_MsgType | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_PUTAPPLTYPE | JMS_IBM_PutApplType | System.Int32 | Send [MessageProducer] |
| JMS_IBM_PUTDATE | JMS_IBM_PutDate | System.String | Send [MessageProducer] |
| JMS_IBM_PUTTIME | JMS_IBM_PutTime | System.String | Send [MessageProducer] |
| JMS_IBM_REPORT_COA | JMS_IBM_Report_COA | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_COD | JMS_IBM_Report_COD | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_DISCARD_MSG | JMS_IBM_Report_Discard_Msg | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_EXCEPTION | JMS_IBM_Report_Exception | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_EXPIRATION | JMS_IBM_Report_Expiration | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_NAN | JMS_IBM_Report_NAN | System.Int32 | Set Integer Property [PropertyContext] |

| 표 *21. IBM* 정의 메시지 특성 *(계속)* | | | |
|---|---|---|---|
| **IBM 정의 특성의 XMS 이름** | **WebSphere JMS 이름** | **데이터 유형** | **전송된 메시지에 대한 값 설정 방법(*method* [*class*] 형식)** |
| JMS_IBM_REPORT_PAN | JMS_IBM_Report_PAN | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_PASS_CORREL_ ID | JMS_IBM_Report_Pass_Correl_ID | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_REPORT_PASS_MSG_ID | JMS_IBM_Report_Pass_Msg_ID | System.Int32 | Set Integer Property [PropertyContext] |
| JMS_IBM_SYSTEM_MESSAGEID | JMS_IBM_System_MessageID | System.String | Send [MessageProducer] |

### 애플리케이션 정의 메시지 특성

XMS 애플리케이션은 자체 메시지 특성 세트를 작성하여 사용할 수 있습니다. 애플리케이션이 메시지를 전송할 때 해당 특성도 메시지와 함께 전송됩니다. 수신 애플리케이션은 메시지 선택자를 사용하여 이러한 특성 값에 기반하여 수신할 메시지를 선택할 수 있습니다.

WebSphere JMS 애플리케이션이 XMS 애플리케이션에서 송신한 메시지를 선택하고 처리하도록 하려면 *Java*를 사용하는 *WebSphere MQ*에 설명된 대로 애플케이션 정의 특성의 이름이 메시지 선택자 표현식에서 ID를 구성하기 위한 규칙을 따라야 합니다. 애플리케이션에서 정의된 특성의 값에는 다음 데이터 유형 중 하나가 있어야 합니다. System.Boolean, System.SByte, System.Int16, System.Int32, System.Int64, System.Float, System.Double 또는 System.String.

## XMS 메시지 본문

메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

XMS는 다섯 가지 유형의 메시지 본문을 지원합니다.

**바이트**
본문에 바이트 스트림이 포함됩니다. 이 유형의 본문이 있는 메시지를 바이트 메시지라고 합니다. IBytesMessage 인터페이스에는 바이트 메시지의 본문을 처리하는 메소드가 포함되어 있습니다. 자세한 정보는 70 페이지의 『바이트 메시지』의 내용을 참조하십시오.

**맵**
본문은 이름-값 쌍 세트를 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다. 이 유형의 본문이 있는 메시지를 맵 메시지라고 합니다. IMapMessage 인터페이스에는 맵 메시지 본문을 처리하는 메소드가 포함되어 있습니다. 자세한 정보는 71 페이지의 『맵 메시지』의 내용을 참조하십시오.

**오브젝트**
본문에는 직렬화된 Java 또는 .NET 오브젝트가 포함되어 있습니다. 이 유형의 본문이 있는 메시지를 오브젝트 메시지라고 합니다. IObjectMessage 인터페이스에는 오브젝트 메시지의 본문을 처리하는 메소드가 있습니다. 자세한 정보는 71 페이지의 『오브젝트 메시지』의 내용을 참조하십시오.

**스트림**
본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다. 이 유형의 본문이 있는 메시지를 스트림 메시지라고 합니다. IStreamMessage 인터페이스에는 스트림 메시지의 본문을 처리하는 메소드가 포함되어 있습니다. 자세한 정보는 72 페이지의 『스트림 메시지』의 내용을 참조하십시오.

**텍스트**
본문에 문자열이 포함됩니다. 이 유형의 본문이 있는 메시지를 텍스트 메시지라고 합니다. ITextMessage 인터페이스에는 텍스트 메시지의 본문을 처리하는 메소드가 포함되어 있습니다. 자세한 정보는 73 페이지의 『텍스트 메시지』의 내용을 참조하십시오.

IMessage 인터페이스는 모든 메시지 오브젝트의 상위이며 메시징 기능에서 XMS 메시지 유형을 나타내는 데 사용될 수 있습니다.

이러한 데이터 유형 각각의 크기, 최대값 및 최소값에 대한 정보는 37 페이지의 표 5의 내용을 참조하십시오.

**관련 참조**

XMS 메시지의 파트
XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

XMS 메시지의 헤더 필드
XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

XMS 메시지의 특성
XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

메시지 선택자
XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

XMS 메시지를 WebSphere MQ 메시지로 맵핑
XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

## 애플리케이션 데이터 요소의 데이터 유형

XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

이러한 이유로, XMS 애플리케이션에서 메시지 본문에 작성된 애플리케이션 데이터의 각 요소에는 69 페이지의 표 22에 나열된 데이터 유형 중 하나가 있어야 합니다. 각 데이터 유형에 대해 표에서는 호환 가능한 Java 데이터 유형을 표시합니다. XMS는 이러한 데이터 유형만으로 애플리케이션 데이터 요소를 작성할 수 있는 메소드를 제공합니다.

표 *22. Java* 데이터 유형과 호환 가능한 *XMS* 데이터 유형

| **XMS 데이터 유형** | **의미** | **호환 가능 Java 데이터 유형** |
|---|---|---|
| System.Boolean | Boolean 값 True 또는 False | 부울 |
| System.Char16 | 2바이트 문자 | char |
| System.SByte | 부호가 있는 8비트 정수 | byte |
| System.Int16 | 부호가 있는 16비트 정수 | 쇼트 |
| System.Int32 | 부호 있는 32비트 정수 | int |
| System.Int64 | 부호가 있는 64비트 정수 | 롱 |
| System.Float | 부호가 있는 부동 소수점 수 | 부동 소수점 |
| System.Double | 부호가 있는 배정밀도 부동 소수점 수 | 실수(double) |
| System.String | 문자열 | 문자열 |

이러한 데이터 유형 각각의 크기, 최대값 및 최소값에 대한 정보는 36 페이지의 『XMS 기본 유형』의 내용을 참조하십시오.

**관련 개념**

오브젝트의 속성 및 특성
XMS 오브젝트에는 여러 가지 방법으로 구현되는 오브젝트 특성인 속성과 특성을 포함할 수 있습니다.

XMS 기본 유형
XMS에서는 동일한 8개의 Java 기본 유형(byte, short, int, long, float, double, char 및 boolean)을 제공합니다. 이를 사용하여 데이터 손실이나 손상 없이 XMS와 JMS 간에 메시지를 교환할 수 있습니다.

한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환

애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

**관련 참조**

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

오브젝트 메시지
오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

## 바이트 메시지

바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

XMS 애플리케이션이 XMS 또는 JMS 애플리케이션 프로그래밍 인터페이스를 사용하지 않는 애플리케이션과 메시지를 교환해야 하는 경우, 바이트 메시지가 유용합니다.

애플리케이션이 바이트 메시지를 작성한 후에는 메시지 본문이 쓰기 전용이 됩니다. 애플리케이션은 .NET에 대해 IBytesMessage 인터페이스의 해당 쓰기 메소드를 호출하여 애플리케이션 데이터를 본문에 어셈블합니다. 애플리케이션이 바이트 메시지 스트림에 값을 쓸 때마다 그 값이 애플리케이션에 의해 작성된 이전 값 바로 뒤에 어셈블됩니다. XMS는 내부 커서를 유지보수하여 어셈블된 마지막 바이트의 위치를 기억합니다.

애플리케이션이 메시지를 전송할 때 메시지의 본문은 읽기 전용이 됩니다. 이 모드에서는 애플리케이션이 반복적으로 메시지를 전송할 수 있습니다.

애플리케이션이 바이트 메시지를 수신할 때 메시지 본문은 읽기 전용입니다. 애플리케이션은 IBytesMessage 인터페이스의 해당 읽기 메소드를 사용하여 바이트 메시지 스트림의 컨텐츠를 읽을 수 있습니다. 애플리케이션은 바이트를 순서대로 읽고 XMS는 내부 커서를 유지보수하여 읽은 마지막 바이트의 위치를 기억합니다.

바이트 메시지 본문이 쓰기 가능한 경우 애플리케이션이 IBytesMessage 인터페이스의 Reset 메소드를 호출하면 본문은 읽기 전용이 됩니다. 또한 메소드가 커서를 바이트 메시지 스트림의 시작으로 옮깁니다.

바이트 메시지 본문이 읽기 전용이면 애플리케이션이 .NET 에 대한 IMessage 인터페이스의 `Clear Body` 메소드를 호출하는 경우 본문이 쓰기 가능하게 됩니다. 또한 메소드가 본문을 지웁니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

오브젝트 메시지
오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

IBytesMessage(.NET 인터페이스의 경우)
바이트 메시지는 본문이 바이트 스트림으로 구성된 메시지입니다.

## 맵 메시지

맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

각 이름-값 쌍에서, 이름은 값을 식별하는 문자열이며 값은 69 페이지의 표 22에 나열된 XMS 데이터 유형 중 하나가 포함된 애플리케이션 데이터의 요소입니다. 이름-값 쌍의 순서는 정의되지 않습니다. MapMessage 클래스에는 이름-값 쌍을 설정하고 가져오는 메소드가 포함됩니다.

애플리케이션은 이름을 지정하여 이름-값 쌍에 임의로 액세스할 수 있습니다.

.NET 애플리케이션은 MapNames 특성을 사용하여 맵 메시지 본문에 있는 이름 나열을 가져올 수 있습니다.

애플리케이션이 이름-값 쌍의 값을 가져올 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 예를 들어 맵 메시지 본문에서 정수를 가져오기 위해 애플리케이션이 MapMessage 클래스의 GetString 메소드를 호출할 수 있습니다. 이는 정수를 문자열로서 리턴합니다. 지원되는 변환은 XMS가 특성 값을 한 데이터 유형에서 다른 유형으로 변환할 때 지원되는 것과 동일합니다. 지원되는 변환에 대한 자세한 정보는 37 페이지의 『한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환』의 내용을 참조하십시오.

애플리케이션이 맵 메시지를 작성한 후에는 메시지 본문이 읽기 및 쓰기가 가능하게 됩니다. 애플리케이션이 메시지를 전송한 후에도 본문은 읽기 및 쓰기가 가능합니다. 애플리케이션이 맵 메시지를 수신할 때 메시지 본문은 읽기 전용입니다. 맵 메시지 본문이 읽기 전용일 때 애플리케이션이 Message 클래스의 Clear Body 메소드를 호출하면 본문에서 읽기 및 쓰기가 가능하게 됩니다. 또한 메소드가 본문을 지웁니다.

**관련 개념**

한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환
애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

오브젝트 메시지
오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

IMapMessage(.NET 인터페이스의 경우)
맵 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 이름-값 쌍 세트로 구성된 메시지입니다.

## 오브젝트 메시지

오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

XMS 애플리케이션은 오브젝트 메시지를 수신하고 해당 헤더 필드와 특성을 변경한 후 다른 대상으로 전송할 수 있습니다. 또한 애플리케이션은 오브젝트 메시지의 본문을 복사하여 다른 오브젝트 메시지를 구성하는 데 사용할 수 있습니다. XMS는 오브젝트 메시지의 본문을 바이트 배열로 취급합니다.

애플리케이션이 오브젝트 메시지를 작성한 후에는 메시지 본문이 읽기 및 쓰기가 가능하게 됩니다. 애플리케이션이 메시지를 전송한 후에도 본문은 읽기 및 쓰기가 가능합니다. 애플리케이션이 오브젝트 메시지를 수신할 때 메시지 본문은 읽기 전용입니다. 애플리케이션이 오브젝트 메시지의 본문이 읽기 전용인 경우 .NET 에 대한 IMessage 인터페이스의 `Clear Body` 메소드를 호출하는 경우 본문을 읽기 및 쓰기 가능하게 됩니다. 또한 메소드가 본문을 지웁니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

IObjectMessage(.NET 인터페이스의 경우)
오브젝트 메시지는 본문이 직렬화된 Java 또는 .NET 오브젝트로 구성된 메시지입니다.

## 스트림 메시지

스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

값의 데이터 유형은 69 페이지의 표 22에 나열된 XMS 데이터 유형 중 하나입니다.

애플리케이션이 스트림 메시지를 작성한 후 메시지 본문은 쓰기 가능합니다. 애플리케이션은 .NET IStreamMessage 인터페이스의 해당 쓰기 메소드를 호출하여 애플리케이션 데이터를 본문에 어셈블합니다. 애플리케이션이 메시지 스트림에 값을 쓸 때마다 그 값과 해당 유형은 애플리케이션에 의해 작성된 이전 값 바로 뒤에 어셈블됩니다. XMS는 내부 커서를 유지보수하여 어셈블된 마지막 값의 위치를 기억합니다.

애플리케이션이 메시지를 전송할 때 메시지의 본문은 읽기 전용이 됩니다. 애플리케이션은 이 모드에서 메시지를 여러 번 전송할 수 있습니다.

애플리케이션이 스트림 메시지를 수신할 때 메시지 본문은 읽기 전용입니다. 애플리케이션은 .NET IStreamMessage 인터페이스의 해당 읽기 메소드를 사용하여 메시지 스트림의 컨텐츠를 읽을 수 있습니다. 애플리케이션은 값을 순서대로 읽고 XMS는 내부 커서를 유지보수하여 읽은 마지막 값의 위치를 기억합니다.

애플리케이션이 메시지 스트림에서 값을 읽을 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 예를 들어 메시지 스트림에서 정수를 읽으려면 애플리케이션은 문자열로 정수를 리턴하는 ReadString 메소드를 호출할 수 있습니다. 지원되는 변환은 XMS가 특성 값을 한 데이터 유형에서 다른 유형으로 변환할 때 지원되는 것과 동일합니다. 지원되는 변환에 대한 자세한 정보는 37 페이지의 『한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환』의 내용을 참조하십시오.

애플리케이션이 메시지 스트림에서 값을 읽으려고 시도할 때 오류가 발생하면 커서가 진행되지 않습니다. 애플리케이션은 값을 다른 데이터 유형으로 읽으려고 시도하여 오류를 복구할 수 있습니다.

스트림 메시지 본문이 쓰기 전용인 경우 애플리케이션이 .NET IStreamMessage 인터페이스의 Reset 메소드를 호출하면 본문은 읽기 전용이 됩니다. 또한 메소드가 커서를 메시지 스트림의 시작으로 옮깁니다.

스트림 메시지 본문이 읽기 전용인 경우 애플리케이션이 .NET 에 대한 IMessage 인터페이스의 `Clear Body` 메소드를 호출하는 경우 본문은 쓰기 전용이 됩니다. 또한 메소드가 본문을 지웁니다.

**관련 개념**

한 데이터 유형에서 다른 데이터 유형으로 특성 값의 암시적 변환
애플리케이션이 특성의 값을 가져오면 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 지원되는 변환 및 XMS의 변환 수행 방법에 적용되는 규칙은 여러 가지가 있습니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

오브젝트 메시지

오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

IStreamMessage(.NET 인터페이스의 경우)
스트림 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 값 스트림으로 구성된 메시지입니다. 본문의 컨텐츠는 순차적으로 쓰여지고 읽힙니다.

## 텍스트 메시지

텍스트 메시지의 본문은 문자열을 포함합니다.

애플리케이션이 텍스트 메시지를 작성한 후에는 메시지 본문이 읽기 및 쓰기가 가능하게 됩니다. 애플리케이션이 메시지를 전송한 후에도 본문은 읽기 및 쓰기가 가능합니다. 애플리케이션이 텍스트 메시지를 수신할 때 메시지 본문은 읽기 전용입니다. 텍스트 메시지 본문이 읽기 전용인 경우 애플리케이션이 .NET IMessage 인터페이스의 Clear Body 메소드를 호출하면 본문은 읽기 가능 및 쓰기 가능이 됩니다. 또한 메소드가 본문을 지웁니다.

**관련 참조**

애플리케이션 데이터 요소의 데이터 유형
XMS 애플리케이션이 JMS용 WebSphere MQ 클래스 애플리케이션과 메시지를 교환할 수 있도록 하려면 두 애플리케이션이 동일한 방식으로 메시지 본문에 있는 애플리케이션 데이터를 해석할 수 있어야 합니다.

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

맵 메시지
맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

오브젝트 메시지
오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

ITextMessage(.NET 인터페이스의 경우)
텍스트 메시지는 본문이 문자열로 구성된 메시지입니다.

## 메시지 선택자

XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

애플리케이션이 메시지 이용자를 작성할 때 메시지 선택자 표현식과 이용자를 연관시킬 수 있습니다. 메시지 선택자 표현식이 선택 기준을 지정합니다.

애플리케이션이 IBM WebSphere MQ V7.0 큐 관리자에 연결될 때 큐 관리자에서 메시지 선택이 수행됩니다. XMS는 아무 것도 선택하지 않고 단순히 큐 관리자로부터 받은 메시지를 전달만 하므로 성능이 개선됩니다.

애플리케이션이 둘 이상의 메시지 이용자를 작성할 수 있고, 각각에는 자체 메시지 선택자 표현식이 있습니다. 수신 메시지가 두 개 이상의 메시지 이용자 선택 기준을 충족시킬 경우 XMS는 해당 이용자 각각에게 메시지를 전달합니다.

메시지 선택자 표현식은 다음과 같은 메시지 특성을 참조할 수 있습니다.

- JMS 정의 특성
- IBM 정의 특성
- 애플리케이션 정의 특성

또한 다음과 같은 메시지 헤더 필드도 참조할 수 있습니다.

- JMSCorrelationID
- JMSDeliveryMode
- JMSMessageID
- JMSPriority

- JMSTimestamp
- JMSType

그러나 메시지 선택자 표현식이 메시지 본문의 데이터는 참조할 수 없습니다.

다음은 메시지 선택자 표현식의 한 예입니다.

```
JMSPriority > 3 AND manufacturer = 'Jaguar' AND model in ('xj6','xj12')
```

XMS 는 메시지가 3보다 큰 우선순위를 갖는 경우에만 이 메시지 선택자 표현식을 사용하는 메시지 처리자에게 메시지를 전달합니다. 애플리케이션 정의 특성, 제조업체, 값이 `Jaguar;` 이고 다른 애플리케이션이 정의한 특성, 모델, 값이 `xj6` 또는 `xj12.` 인 모델

XMS의 메시지 선택자 표현식 형성 구문 규칙은 JMS용 WebSphere MQ 클래스의 구문 규칙과 동일합니다. 메시지 선택자 표현식 구성에 대한 정보는 *Java*를 사용하는 *WebSphere MQ*를 참조하십시오. 메시지 선택자 표현식에서 JMS 정의 특성의 이름은 JMS 이름이어야 하며, IBM 정의 특성의 이름은 JMS용 WebSphere MQ 클래스 이름이어야 합니다. 메시지 선택자 표현식에서 XMS 이름을 사용할 수 없습니다.

**관련 참조**

XMS 메시지의 파트
XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

XMS 메시지의 헤더 필드
XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

XMS 메시지의 특성
XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

XMS 메시지 본문
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

XMS 메시지를 WebSphere MQ 메시지로 맵핑
XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

## XMS 메시지를 WebSphere MQ 메시지로 맵핑

XMS 메시지의 JMS 헤더 필드 및 특성이 WebSphere MQ 메시지의 헤더 구조에 있는 필드에 맵핑됩니다.

XMS 애플리케이션이 WebSphere MQ 큐 관리자에 연결되는 경우, 유사한 환경에서 JMS용 WebSphere MQ 클래스 메시지가 WebSphere MQ 메시지에 맵핑되는 방식과 동일한 방법으로 큐 관리자에 전송된 메시지를 WebSphere MQ 메시지로 맵핑합니다.

Destination 오브젝트의 XMSC_WMQ_TARGET_CLIENT 특성이 XMSC_WMQ_TARGET_DEST_JMS로 설정되는 경우 목적지로 전송된 메시지의 특성 및 JMS 헤더 필드는 WebSphere MQ 메시지의 MQMD 및 MQRFH2 헤더 구조에 있는 필드로 맵핑됩니다. 이런 방법으로 XMSC_WMQ_TARGET_CLIENT 특성을 설정할 경우 메시지를 수신하는 애플리케이션이 MQRFH2 헤더를 처리할 수 있다고 가정합니다. 따라서 수신 애플리케이션은 또 다른 XMS 애플리케이션인 JMS용 WebSphere MQ 클래스 애플리케이션 또는 MQRFH2 헤더를 핸들링하도록 설계된 기본 WebSphere MQ 애플리케이션일 수 있습니다.

Destination 오브젝트의 XMSC_WMQ_TARGET_CLIENT 특성이 대신 XMSC_WMQ_TARGET_DEST_MQ로 설정되는 경우에는 목적지로 전송된 메시지의 특성 및 JMS 헤더 필드가 WebSphere MQ 메시지의 MQMD 헤더 구조에 있는 필드로 맵핑됩니다. 이 메시지에는 MQRFH2 헤더가 없으며, MQMD 헤더 구조의 필드로 맵핑할 수 없는 모든 JMS 헤더 필드 및 특성은 무시됩니다. 그러므로 메시지를 수신하는 애플리케이션은 MQRFH2 헤더를 처리하도록 디자인되지 않은 고유 WebSphere MQ일 수 있습니다.

큐 관리자에서 수신된 WebSphere MQ 메시지는 비슷한 상황에서 WebSphere MQ 메시지가 JMS용 WebSphere MQ 클래스 메시지에 맵핑되는 방법과 동일한 방식으로 XMS 메시지에 맵핑됩니다.

수신 WebSphere MQ 메시지에 MQRFH2 헤더가 있는 경우 결과 XMS 메시지의 본문은 MQRFH2 헤더의 `mcd` 폴더에 포함된 **Msd** 등록 정보의 값으로 판별됩니다. **Msd** 특성이 MQRFH2 헤더에 없거나 WebSphere MQ 메시지에 MQRFH2 헤더에 없는 경우, 결과적으로 XMS 메시지에는 MQMD 헤더의 *Format* 필드 값으로 유형이 판별되

는 본문이 있습니다. *Format* 필드가 MQFMT_STRING으로 설정된 경우 XMS 메시지는 텍스트 메시지입니다. 그렇지 않으면, XMS 메시지는 바이트 메시지입니다. WebSphere MQ 메시지에 MQRFH2 헤더가 없을 경우 MQMD 헤더의 필드에서 파생할 수 있는 JMS 헤더 필드 및 특성만 설정됩니다.

For more information about mapping JMS용 WebSphere MQ 클래스 messages onto WebSphere MQ messages, see *WebSphere MQ Java* 사용.

**관련 참조**

XMS 메시지의 파트
XMS 메시지는 헤더, 특성 세트 및 본문으로 구성됩니다.

XMS 메시지의 헤더 필드
XMS 애플리케이션이 WebSphere JMS 애플리케이션과 메시지를 교환할 수 있도록 XMS 메시지의 헤더에 JMS 메시지 헤더 필드를 포함합니다.

XMS 메시지의 특성
XMS는 세 종류의 메시지 특성을 지원합니다. JMS 정의 특성, IBM 정의 특성 및 애플리케이션 정의 특성

XMS 메시지 본문
메시지 본문에는 애플리케이션 데이터가 들어 있습니다. 그러나 메시지에는 본문이 포함될 수 없으며 헤더 필드 및 특성만으로 구성됩니다.

메시지 선택자
XMS 애플리케이션은 수신하려는 메시지를 선택하기 위해 메시지 선택자를 사용합니다.

### *Message Service Client for .NET* 애플리케이션에서 메시지 디스크립터 읽기 및 쓰기

StrucId 및 Version을 제외한 WebSphere MQ 메시지의 모든 메시지 디스크립터(MQMD) 필드에 액세스할 수 있습니다. BackoutCount는 읽을 수 있지만 쓰기 작업은 수행할 수 없습니다. 이 기능은 WebSphere MQ 큐 관리자 버전 6 이상에 연결하는 경우에만 사용할 수 있고 나중에 설명하는 대상 특성에서 제어됩니다.

Message Service Client for .NET에서 제공하는 메시지 속성을 사용하면 XMS 애플리케이션이 MQMD 필드를 설정하고 WebSphere MQ 애플리케이션을 구동하기도 쉬워집니다.

발행/구독 메시징을 사용할 때 일부 제한사항이 적용됩니다. 예를 들어 MsgID 및 CorrelId와 같은 MQMD 필드는 무시됩니다.

WebSphere MQ V6 큐 관리자에 연결하는 경우, 이 주제에 설명된 기능을 발행/구독 메시징에 사용할 수 없습니다. 이는 **PROVIDERVERSION** 특성이 6으로 설정되는 경우에도 사용할 수 없습니다.

### *Message Service Client for .NET* 애플리케이션에서 *IBM WebSphere MQ* 메시지 데이터에 액세스

JMSBytesMessage의 본문으로 Message Service Client for .NET 애플리케이션의 MQRFH2 헤더(있는 경우) 및 다른 WebSphere MQ 헤더(있는 경우)를 포함하여 전체 WebSphere MQ 메시지 데이터에 액세스할 수 있습니다.

이 주제에 설명된 기능은 버전 7 이상의 WebSphere MQ 큐 관리자에 연결하고 WebSphere MQ 메시지 제공자가 정상 모드인 경우에만 사용할 수 있습니다.

대상 오브젝트 특성은 XMS 애플리케이션에서 JMSBytesMessage의 본문으로 WebSphere MQ 메시지 전체(MQRFH2 포함. 있는 경우)에 액세스하는 방식을 결정합니다.

## 문제점 해결

이 장에서는 Message Service Client for .NET를 사용할 때 문제점을 발견하고 해결하는 데 도움이 되는 정보를 제공합니다.

이 장에는 다음 절가 포함됩니다.

- 76 페이지의 『.NET 애플리케이션에 대한 추적 구성』
- 79 페이지의 『.NET 애플리케이션에 대한 FFDC 구성』
- 80 페이지의 『문제점 해결 팁』

## .NET 애플리케이션에 대한 추적 구성

XMS .NET 애플리케이션의 경우 XMS 환경 변수뿐 아니라 애플리케이션 구성 파일에서도 추적을 구성할 수 있습니다. 추적할 컴포넌트를 선택할 수 있습니다. 추적은 일반적으로 IBM 지원 센터의 안내에 따라 사용됩니다.

XMS .NET에 대한 추적은 표준 .NET 추적 인프라스트럭처를 기반으로 합니다.

오류 추적을 제외한 모든 추적은 기본적으로 사용 불가능합니다. 추적을 켜고 다음 방법 중 하나로 추적 설정을 구성할 수 있습니다.

- 파일과 관련된 실행 가능 프로그램의 이름 및 접미부 `.config`로 구성된 이름의 애플리케이션 구성 파일 사용. 예를 들어 text.exe에 대한 애플리케이션 구성 파일의 이름은 text.exe.config가 됩니다. 애플리케이션 구성 파일의 사용은 XMS .NET 애플리케이션에 대한 추적을 사용 가능하게 하는 좋은 방법입니다. 자세한 정보는 77 페이지의 『애플리케이션구성 파일을 사용하여 추적 구성』의 내용을 참조하십시오.
- XMS C 또는 C++ 애플리케이션에 XMS 환경 변수 사용. 자세한 정보는 78 페이지의 『XMS 환경 변수를 사용하여 추적 구성』의 내용을 참조하십시오.

The active trace file has a name of the format xms_trace<*PID*>.log where <*PID*> represents the process ID of the application. 활성 추적 파일의 크기는 기본적으로 20MB로 제한됩니다. 이 제한에 도달하면 파일은 이름이 바뀌고 아카이브됩니다. Archived files have names of the format xms_trace<*PID*>*_YY.MM.DD_HH.MM.SS*.log

기본적으로 보관된 추적 파일 수는 4(활성 파일 1 및 아카이브 파일 3)입니다. 이 네 개의 파일은 애플리케이션이 중지할 때까지 롤링 버퍼로 사용되며 가장 오래된 파일이 제거되고 새 파일로 바뀝니다. 애플리케이션 구성 파일에서 다른 숫자를 지정하여 추적 파일 수를 변경할 수 있습니다. 그러나 파일은 둘 이상이어야 합니다(하나의 활성 파일과 하나의 아카이브된 파일).

두 가지 추적 파일 형식을 사용할 수 있습니다.

- 기본 형식 추적 파일은 사람이 읽을 수 있는 WebSphere 애플리케이션 서버 형식으로 되어 있습니다. 이 형식은 기본 추적 파일 형식입니다. 기본 형식은 추적 분석기 도구와 호환되지 않습니다.
- 고급 형식 추적 파일은 추적 분석기 도구와 호환됩니다. 추적 파일을 고급 형식으로 생성할지를 애플리케이션 구성 파일에 지정해야 합니다.

추적 항목에 포함되는 정보는 다음과 같습니다.

- 추적을 로깅하는 날짜 및 시간
- 클래스 이름
- 추적 유형
- 추적 메시지

다음 예에서는 일부 추적의 추출을 보여줍니다.

```
[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
```

이전 예에서 형식은 다음과 같습니다.

```
[Date Time:Microsecs]         Thread-id    Classname        Trace-type    Methodname    Entry
or Exit
```

여기서, `Trace-type`은 다음과 같습니다.

>: 입력
<: 종료
d: 디버그 정보

## 애플리케이션구성 파일을 사용하여 추적 구성

XMS .NET 애플리케이션에 대한 추적을 구성하는 좋은 방법은 애플리케이션 구성 파일을 사용하는 것입니다. 이 파일의 trace 섹션에는 추적할 내용을 정의하는 매개변수, 추적 파일 위치 및 허용되는 최대 크기, 사용되는 추적 파일 수, 추적 파일 형식이 포함됩니다.

애플리케이션 구성 파일을 사용하여 추적을 켜려면, 애플리케이션의 실행 파일과 같은 디렉토리에 파일을 놓습니다.

추적은 컴포넌트와 추적 유형별로 사용 가능하게 할 수 있습니다. 또한 전체 추적 그룹에 대한 추적을 켤 수도 있습니다. 개별적으로 또는 전체적으로 계층 구조에서 컴포넌트에 대한 추적을 켤 수 있습니다. 사용 가능한 추적의 유형은 다음과 같습니다.

- 디버그 추적
- 예외 추적
- 경고, 정보용 메시지, 오류 메시지
- 메소드 입력 및 종료 추적

다음 예에서는 애플리케이션 구성 파일의 Trace 섹션에 정의된 추적 설정을 보여줍니다.

```
<?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>
```

77 페이지의 표 23에서는 매개변수 설정을 자세히 설명합니다.

표 *23.* 애플리케이션 구성 파일 추적 매개변수 설정

| **매개변수** | **설명** |
|---|---|
| traceSpecification=<ComponentName>=<type>=<state> | <ComponentName>은 추적할 클래스의 이름입니다. 이 이름에 * 와일드 카드 문자를 사용할 수 있습니다. 예를 들어, *=all=enabled는 모든 클래스를 추적하도록 지정하고 IBM.XMS.impl.*=all=enabled는 필요한 API 추적만 지정합니다.<br><br><type>은 다음 추적 유형 중 하나입니다.<br>• 모두<br>• debug<br>• 이벤트<br>• EntryExit<br><br><state>는 사용 또는 사용 안함일 수 있습니다.<br><br>':'(콜론) 분리문자로 구분하여 여러 추적 요소를 함께 사용할 수 있습니다. |

| 표 *23.* 애플리케이션 구성 파일 추적 매개변수 설정 *(계속)* | |
|---|---|
| **매개변수** | **설명** |
| `traceFilePath="<filename>"` | traceFilePath를 지정하지 않거나 traceFilePath가 있지만 빈 문자열이 포함된 경우 추적 파일은 현재 디렉토리에 저장됩니다. 추적 파일을 이름 지정된 디렉토리에 저장하려면 traceFilePath에 디렉토리 이름을 지정하십시오. 예를 들어, 다음과 같습니다.<br>`traceFilePath="c:\somepath"` |
| `traceFileSize="<size>"` | 추적 파일의 최대 허용 크기입니다. 파일이 이 크기가 되면 아카이브되고 이름이 바뀝니다. 기본 최대값은 20KB입니다. 이는 `traceFileSize="20000000"`으로 지정됩니다. |
| `traceFileNumber="<number>"` | 보관할 추적 파일 수입니다. 기본값은 4입니다(활성 파일 1 및 아카이브 파일 3). 허용되는 최소값은 2입니다. |
| `traceFormat="<format>"` | 기본 추적 형식은 basic입니다. `traceFormat="basic"`을 지정하거나 traceFormat을 지정하지 않거나 traceFormat이 있지만 빈 문자열을 포함하는 경우 이 형식으로 추적 파일이 생성됩니다.<br>추적 분석기 도구와 호환 가능한 추적이 필요한 경우 `traceFormat="advanced"`를 지정해야 합니다. |

애플리케이션 구성 파일의 추적 설정값은 동적이므로 파일이 저장되거나 바뀔 때마다 다시 읽힙니다. 편집한 후에 파일에서 오류가 발견되면 추적 파일 설정은 기본값으로 돌아갑니다.

**관련 개념**

XMS 환경 변수를 사용하여 추적 구성

애플리케이션 구성 파일을 사용하는 대신, XMS 환경 변수를 사용하여 추적을 켤 수 있습니다. 이 환경 변수는 애플리케이션 구성 파일에 추적 스펙이 없는 경우에만 사용됩니다.

## XMS 환경 변수를 사용하여 추적 구성

애플리케이션 구성 파일을 사용하는 대신, XMS 환경 변수를 사용하여 추적을 켤 수 있습니다. 이 환경 변수는 애플리케이션 구성 파일에 추적 스펙이 없는 경우에만 사용됩니다.

XMS .NET 애플리케이션에 대한 추적을 구성하려면 애플리케이션을 실행하기 전에 다음 환경 변수를 설정하십시오.

| 표 *24. .NET* 추적을 사용할 환경 변수 설정 | | | |
|---|---|---|---|
| **환경 변수** | **기본값** | **설정** | **의미** |
| XMS_TRACE_ON | 적용할 수 없음 | 적용 불가능: 이 변수의 값은 무시됨 | XMS_TRACE_ON이 설정되면 기본적으로 모든 추적이 사용 가능합니다. |

표 *24. .NET* 추적을 사용할 환경 변수 설정 *(계속)*

| 환경 변수 | 기본값 | 설정 | 의미 |
|---|---|---|---|
| XMS_TRACE_FILE_PATH | Current® 작업 디렉토리 | /dirpath/ | 추적과 FFDC 레코드가 쓰여지는 디렉토리 경로.<br>XMS에서는 대체 위치를 지정하지 않는 한, 현재 작업 디렉토리에 FFDC 및 추적 파일을 작성합니다. XMS_TRACE_FILE_PATH 환경 변수를 XMS에서 FFDC 및 추적 파일을 작성하려는 디렉토리의 완전한 경로 이름으로 설정하여 대체 위치를 지정할 수 있습니다. 추적하려는 애플리케이션을 시작하기 전에 이 환경 변수를 설정해야 합니다. 애플리케이션을 실행하는 사용자 ID가 XMS에 의해 FFDC 및 추적 파일이 작성되는 디렉토리에 대한 쓰기 권한을 갖는지 확인해야 합니다. |
| XMS_TRACE_FORMAT | 기본 | BASIC, ADVANCED | 필수 추적 형식을 지정합니다(BASIC 또는 ADVANCED). 기본 형식은 BASIC입니다. ADVANCED 형식은 추적 분석기 도구와 호환됩니다. |
| XMS_TRACE_SPECIFICATION | 적용할 수 없음 | 77 페이지의 『애플리케이션구성 파일을 사용하여 추적 구성』 참조 | 77 페이지의 『애플리케이션구성 파일을 사용하여 추적 구성』에 지정된 형식을 따르는 추적 스펙을 대체합니다. |

**관련 개념**

애플리케이션구성 파일을 사용하여 추적 구성

XMS .NET 애플리케이션에 대한 추적을 구성하는 좋은 방법은 애플리케이션 구성 파일을 사용하는 것입니다. 이 파일의 trace 섹션에는 추적할 내용을 정의하는 매개변수, 추적 파일 위치 및 허용되는 최대 크기, 사용되는 추적 파일 수, 추적 파일 형식이 포함됩니다.

## .NET 애플리케이션에 대한 FFDC 구성

XMS의 .NET 구현에서는 각 FFDC에 하나의 FFDC 파일이 생성됩니다.

FFDC(First Failure Data Capture) 파일은 사람이 읽을 수 있는 텍스트 파일로 저장됩니다. 이 파일 이름은 `xmsffdc<`*`processID`*`>_<`*`Date`*`>T<`*`Timestamp`*`>.txt` 양식을 사용합니다. 파일 이름의 예로 `xmsffdc264_2006.01.06T13.18.52.990955.txt`를 들 수 있습니다. 시간소인은 마이크로초 단위를 포함합니다.

파일은 예외가 발생한 날짜 및 시간에 시작되고, 뒤에 예외 유형이 나옵니다. 파일은 고유한 짧은 프로브 ID를 포함하며, 이 FFDC가 나타난 위치를 찾을 때 사용할 수 있습니다.

FFDC를 켜기 위해 구성을 수행하지 않아도 됩니다. 기본적으로 모든 FFDC 파일은 현재 디렉토리에 작성됩니다. 그러나 필요한 경우 애플리케이션 구성 파일의 Trace 섹션에서 ffdcDirectory를 변경하여 다른 디렉토리를 지정할 수도 있습니다. 다음 예에서 모든 추적 파일은 c:client\ffdc: 디렉토리에 로그됩니다.

```
<IBM.XMS>
       <Trace ffdc=true ffdcDirectory="c:\client\ffdc"/>
</IBM.XMS>
```

애플리케이션 구성 파일의 Trace 추적에서 FFDC를 false로 설정하여 추적을 사용 안함으로 설정할 수 있습니다.

애플리케이션 구성 파일을 사용하지 않으면 FFDC가 설정되고 추적은 해제됩니다.

# 문제점 해결 팁

이러한 팁을 사용하여 XMS 사용에 대한 문제점을 해결할 수 있습니다.

## XMS 애플리케이션에서 큐 관리자(MQRC_NOT_AUTHORIZED)에 액세스할 수 없습니다.

XMS .NET 클라이언트에는 WebSphere MQ JMS 클라이언트의 동작과 다른 동작이 있을 수 있습니다. 따라서 JMS 애플리케이션에서 할 수 있음에도 XMS 애플리케이션이 큐 관리자에 연결할 수 없음을 발견할 수 있습니다.

- 이 문제점에 대한 간단한 해결 방법은 최대 길이가 12자이고 큐 관리자의 권한 목록에서 완전히 권한 부여된 사용자 ID를 사용하는 것입니다. 이 해결 방법이 적합하지 않은 경우 사용할 수 있는 더 복잡한 다른 방법은 보안 종료를 사용하는 방법입니다. 이 문제에 추가 도움이 필요한 경우 IBM 지원 센터에 문의하십시오.
- 연결 팩토리의 XMSC_USERID 특성을 설정한 경우 로그온한 사용자의 사용자 ID 및 비밀번호와 일치해야 합니다. 이 특성을 설정하지 않은 경우 큐 관리자는 기본적으로 로그온한 사용자의 사용자 ID를 사용합니다.
- WebSphere MQ에 대한 사용자 인증은 XMSC.USERID 및 XMSC.PASSWORD 필드에 제공된 정보가 아니라, 현재 로그온한 사용자의 세부사항을 사용하여 수행합니다. 이는 WebSphere MQ에서 일관성을 유지보수하기 위해 설계되었습니다. 인증에 대한 자세한 정보를 보려면 온라인 IBM IBM WebSphere MQ 제품 문서에서 인증 정보를 참조하십시오.

## 메시징 엔진으로 경로 재지정되는 연결

WebSphere Application Server 버전 6.0.2 서비스 통합 버스에 연결된 경우 모든 연결은 원래 제공자 엔드 포인트에서 클라이언트 연결에 대해 버스가 선택한 메시징 엔진으로 경로 재지정될 수 있습니다. 그럴 경우, 항상 연결이 IP 주소가 아닌, 호스트 이름이 지정한 호스트 서버로 연결을 리디렉션합니다. 따라서 호스트 이름을 분석할 수 없는 경우 연결 문제를 겪을 수 있습니다.

WebSphere Application Server 버전 6.0.2 서비스 통합 버스에 연결하려면 클라이언트 호스트 시스템에서 호스트 이름과 IP 주소 간의 맵핑을 제공해야 합니다. 예를 들어, 클라이언트 호스트 시스템의 로컬 호스트 테이블에서 맵핑을 지정할 수 있습니다.

## Telnet과 유사한 비밀번호 인증에 대한 지원

XMS .NET Real Time Transport 프로토콜은 Telnet과 유사한 비밀번호 인증만 지원합니다. XMS .NET Real Time Transport 프로토콜은 QoP(Quality of Protection)를 지원하지 않습니다.

## 특성 유형 double에 대한 값 설정

Windows 64비트 플랫폼에서, 값이 Double.Epsilon보다 작은 경우 특성 유형 Double의 값을 설정하거나 사용할 때 SetDoubleProperty() 또는 GetDoubleProperty() 메소드가 작동하지 않을 수 있습니다.

예를 들어, Double 유형을 사용하는 특성에 대해 4.9E-324 값을 설정하는 경우, Windows 64비트 플랫폼이 이를 0.0으로 처리합니다. 따라서 분산 메시징 환경에서 JMS 또는 다른 애플리케이션이 Unix 또는 Windows 32비트 시스템에서 Double 특성의 값을 4.9E-324로 설정하고 XMS .Net이 64비트 시스템에서 실행하는 경우, GetDoubleProperty()에서 리턴되는 값은 0.0입니다. 이는 Microsoft .NET 2.0 Framework에서 알려진 문제점입니다.

# .NET용 메시지 서비스 클라이언트 참조

이 참조 절에서는 .NET용 메시지 서비스 클라이언트를 사용하는 데 도움이 되는 정보를 제공합니다. 이 정보는 XMS를 사용한 프로그램밍에 포함된 작업을 수행하는 데 도움을 줍니다.

## .NET 인터페이스

이 절에서는 .NET 클래스 인터페이스와 해당 특성 및 메소드에 대해 설명합니다.

다음 표는 `IBM.XMS` 네임스페이스에 정의된 모든 인터페이스를 요약하여 설명합니다.

표 *25. .NET* 클래스 인터페이스의 요약

| 인터페이스 | 설명 |
|---|---|
| 83 페이지의 『IBytesMessage』 | 바이트 메시지는 본문이 바이트 스트림으로 구성된 메시지입니다. |
| 93 페이지의 『IConnection』 | Connection 오브젝트는 메시징 서버에 대한 애플리케이션의 활성 연결을 나타냅니다. |
| 96 페이지의 『IConnectionFactory』 | 애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다. |
| 98 페이지의 『IConnectionMetaData』 | ConnectionMetaData 오브젝트는 연결에 대한 정보를 제공합니다. |
| 99 페이지의 『IDestination』 | 목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다. |
| 100 페이지의 『ExceptionListener』 | 애플리케이션이 예외 리스너를 사용하여 연결 문제점을 비동기적으로 알립니다. |
| 101 페이지의 『IllegalStateException』 | XMS는 애플리케이션이 잘못되거나 부적절한 시간에 메소드를 호출하는 경우 또는 XMS가 요청된 조작에 적합한 상태가 아닌 경우, 이 예외를 처리합니다. |
| 101 페이지의 『InitialContext』 | 애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다. |
| 103 페이지의 『InvalidClientIDException』 | XMS 는 애플리케이션이 연결에 대한 클라이언트 ID를 설정하려고 시도하지만 클라이언트 ID가 유효하지 않거나 이미 사용 중인 경우 이 예외를 처리합니다. |
| 104 페이지의 『InvalidDestinationException』 | XMS는 애플리케이션에서 유효하지 않은 목적지를 지정한 경우에 이 예외를 처리합니다. |
| 104 페이지의 『InvalidSelectorException』 | XMS는 애플리케이션에서 구문이 유효하지 않은 메시지 선택자 표현식을 제공할 경우에 이 예외를 처리합니다. |
| 104 페이지의 『IMapMessage』 | 맵 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 이름-값 쌍 세트로 구성된 메시지입니다. |
| 113 페이지의 『IMessage』 | Message 오브젝트는 애플리케이션이 송신하거나 수신하는 메시지를 나타냅니다. IMessage는 IMapMessage와 같은 메시지 클래스의 수퍼클래스입니다. |
| 120 페이지의 『IMessageConsumer』 | 애플리케이션에서 메시지 이용자를 사용하여 목적지로 송신된 메시지를 수신합니다. |

| 표 25. .NET 클래스 인터페이스의 요약 (계속) | |
|---|---|
| **인터페이스** | **설명** |
| 122 페이지의 『MessageEOFException』 | XMS 는 애플리케이션이 바이트 메시지 본문을 읽을 때 XMS 가 바이트 메시지 스트림의 끝을 발견하는 경우 이 예외를 처리합니다. |
| 122 페이지의 『MessageFormatException』 | XMS XMS 이 (가) 유효하지 않은 형식의 메시지가 발견되면 이 예외가 발생합니다. |
| 123 페이지의 『IMessageListener(위임)』 | 애플리케이션이 메시지 리스너를 사용하여 메시지를 비동기적으로 수신합니다. |
| 123 페이지의 『MessageNotReadableException』 | 애플리케이션에서 쓰기 전용인 메시지 본문을 읽으려고 하는 경우 XMS에서 이 예외를 처리합니다. |
| 123 페이지의 『MessageNotWritableException』 | XMS 는 애플리케이션이 읽기 전용인 메시지 본문에 쓰려고 시도하는 경우 이 예외를 처리합니다. |
| 124 페이지의 『IMessageProducer』 | 애플리케이션은 메시지 작성자를 사용하여 메시지를 목적지에 송신합니다. |
| 129 페이지의 『IObjectMessage』 | 오브젝트 메시지는 본문이 직렬화된 Java 또는 .NET 오브젝트로 구성된 메시지입니다. |
| 130 페이지의 『IPropertyContext』 | IPropertyContext는 특성을 가져오고 설정하는 메소드를 포함하는 추상 수퍼클래스입니다. 이러한 메소드는 다른 클래스에서 상속합니다. |
| 139 페이지의 『IQueueBrowser』 | 애플리케이션이 큐 브라우저를 사용하여 메시지를 제거하지 않고 큐에서 메시지를 찾습니다. |
| 141 페이지의 『Requestor』 | 애플리케이션이 요청자를 사용하여 요청 메시지를 보내고 잠시 대기한 후 응답을 수신합니다. |
| 143 페이지의 『ResourceAllocationException』 | XMS가 메소드에서 요구하는 자원을 할당할 수 없는 경우 XMS는 이 예외를 처리합니다. |
| 143 페이지의 『SecurityException』 | XMS는 애플리케이션 인증을 위해 제공한 사용자 ID와 비밀번호가 거부된 경우 이 예외를 처리합니다. XMS는 권한 검사에 실패하여 메소드가 완료되지 않은 경우에도 이 예외를 발생시킵니다. |
| 143 페이지의 『ISession』 | 세션은 메시지 송신 및 수신을 위한 단일 스레드 컨텍스트입니다. |
| 154 페이지의 『IStreamMessage』 | 스트림 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 값 스트림으로 구성된 메시지입니다. |
| 163 페이지의 『ITextMessage』 | 텍스트 메시지는 본문이 문자열로 구성된 메시지입니다. |
| 164 페이지의 『TransactionInProgressException』 | XMS 트랜잭션이 진행 중이므로 애플리케이션이 유효하지 않은 조작을 요청하는 경우 이 예외가 발생합니다. |
| 164 페이지의 『TransactionRolledBackException』 | 애플리케이션이 Session.commit()를 호출하여 현재 트랜잭션을 커미트하지만, 트랜잭션이 롤백된 경우 XMS는 이 예외를 처리합니다. |
| XMSC | .NET의 경우 XMS 특성 이름 및 값은 이 클래스에서 공용 상수로 정의됩니다. 자세한 정보는 167 페이지의 『XMS 오브젝트 특성』의 내용을 참조하십시오. |

| 표 25. .NET 클래스 인터페이스의 요약 (계속) | |
|---|---|
| **인터페이스** | **설명** |
| 165 페이지의 『XMSException』 | XMS가 .NET 메소드에 대한 호출을 처리하는 중 오류를 발견한 경우 XMS는 오류를 처리합니다. 예외는 오류 정보를 캡슐화하는 오브젝트입니다.<br>다양한 유형의 XMS 예외가 있으며 XMSException 오브젝트는 이 중 한 가지 예외 유형입니다. 그러나 XMSException 클래스는 다른 XMS 예외 클래스의 수퍼클래스입니다. XMS는 다른 유형의 예외가 적합하지 않은 경우에 XMSException 오브젝트를 처리합니다. |
| 165 페이지의 『XMSFactoryFactory』 | 애플리케이션이 관리 오브젝트를 사용하지 않는 경우 이 클래스를 사용하여 연결 팩토리, 큐 및 토픽을 작성합니다. |

각 메소드의 정의는 XMS가 메소드에 대한 호출을 처리하는 동안 오류를 발견할 경우 리턴되는 예외 코드를 나열합니다. 각 예외 코드는 이름 지정된 상수로 표시됩니다. 이 상수는 해당 예외를 가지고 있습니다.

**관련 개념**

자체 애플리케이션 빌드
샘플 애플리케이션을 빌드하는 것처럼 자체 애플리케이션을 빌드할 수 있습니다.

XMS 애플리케이션 작성
이 장에서는 XMS 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

XMS .NET 애플리케이션 작성
이 장에서는 XMS.NET 애플리케이션을 작성할 때 도움이 되는 정보를 제공합니다.

**관련 참조**

XMS 오브젝트 특성
이 장에서는 XMS에 의해 정의된 오브젝트 특성에 대해 설명합니다.

## IBytesMessage

바이트 메시지는 본문이 바이트 스트림으로 구성된 메시지입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IBytesMessage
```

**관련 참조**

바이트 메시지
바이트 메시지의 본문은 바이트 스트림을 포함합니다. 본문에는 실제 데이터만 있으며 이 데이터를 해석하는 애플리케이션을 전송하고 수신할 책임이 있습니다.

### *.NETproperties*

*BodyLength* - 본문 길이 가져오기

**인터페이스:**

```
Int64 BodyLength
    {
      get;
    }
```

메시지 본문이 읽기 전용일 경우 메시지 본문의 길이를 바이트 단위로 가져옵니다.

메시지 읽기 커서의 현재 위치에 상관없이 리턴값은 전체 본문의 길이입니다.

**예외:**

- XMSException
- MessageNotReadableException

## 방법

*ReadBoolean* - 부울 값 읽기

**인터페이스:**

```
Boolean ReadBoolean();
```

바이트 메시지 스트림에서 부울 값을 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 부울 값입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadSignedByte* - 바이트 읽기

**인터페이스:**

```
Int16   ReadSignedByte();
```

바이트 메시지 스트림의 다음 바이트를 부호 있는 8비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 바이트입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadBytes* - 바이트 읽기

**인터페이스:**

```
Int32   ReadBytes(Byte[] array);
Int32   ReadBytes(Byte[] array, Int32 length);
```

바이트 메시지 스트림에서 현재 커서 위치에서 시작하는 바이트 배열을 읽습니다.

**매개변수:**

**array (output)**
읽은 바이트 배열을 포함하는 버퍼입니다. 호출하기 전에 스트림에서 읽을 나머지 바이트 수가 버퍼 길이보다 크거나 같을 경우 버퍼가 가득 채워집니다. 그렇지 않을 경우 버퍼가 나머지 모든 바이트로 일부만 채워집니다.

입력 시 널 포인터를 지정하면 메소드가 바이트를 읽지 않고 건너뜁니다. 호출하기 전에 스트림에서 읽을 나머지 바이트 수가 버퍼 길이보다 크거나 같을 경우 건너뛴 바이트 수는 버퍼 길이와 같습니다. 그렇지 않을 경우 나머지 모든 바이트를 건너뜁니다. 커너서는 바이트 메시지 스트림의 다음 위치에 남아 있습니다.

**length (input)**
바이트 단위의 버퍼 길이입니다.

**리턴값:**
버퍼로 읽어 들인 바이트 수입니다. 버퍼의 일부만 채워진 경우 값이 버퍼 길이보다 작으며, 이는 더 이상 읽을 바이트가 남아 있지 않음을 나타냅니다. 호출하기 전에 스트림에서 읽을 나머지 바이트가 없을 경우 값은 XMSC_END_OF_STREAM입니다.

입력 시 널 포인터를 지정하면 이 메소드가 값을 리턴하지 않습니다.

**예외:**

- XMSException
- MessageNotReadableException

*ReadChar* - 문자 읽기

**인터페이스:**

```
Char    ReadChar();
```

바이트 메시지 스트림의 다음 2바이트를 문자로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 문자입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadDouble* - 배 정밀도 부동 소수점 수 읽기

**인터페이스:**

```
Double  ReadDouble();
```

바이트 메시지 스트림의 다음 8바이트를 배 정밀도 부동 소수점 수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 배 정밀도 부동 소수점 수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadFloat* - 부동 소수점 수 읽기

**인터페이스:**

```
Single  ReadFloat();
```

바이트 메시지 스트림의 다음 4바이트를 부동 소수점 수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 부동 소수점 수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 정수 읽기

**인터페이스:**

```
Int32   ReadInt();
```

바이트 메시지 스트림의 다음 4바이트를 부호 있는 32비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadLong* - *Long* 정수 읽기

**인터페이스:**

```
Int64   ReadLong();
```

바이트 메시지 스트림의 다음 8바이트를 부호 있는 64비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 Long 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadShort - Short* 정수 읽기

**인터페이스:**

```
Int16   ReadShort();
```

바이트 메시지 스트림의 다음 2바이트를 부호 있는 16비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 Short 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - 부호 없는 바이트 읽기

**인터페이스:**

```
Byte    ReadByte();
```

바이트 메시지 스트림의 다음 바이트를 부호 없는 8비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 바이트입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadUnsignedShort* - 부호 없는 *Short* 정수 읽기

**인터페이스:**

```
Int32   ReadUnsignedShort();
```

바이트 메시지 스트림의 다음 2바이트를 부호 없는 16비트 정수로 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 부호 없는 Short 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadUTF - UTF* 문자열 읽기

**인터페이스:**

```
String  ReadUTF();
```

바이트 메시지 스트림에서 UTF-8로 인코딩된 문자열을 읽습니다.

**참고:** ReadUTF()를 호출하기 전에 버퍼의 커서가 바이트 메시지 스트림의 시작 지점에 있는지 확인하십시오.

**매개변수:**
없음

**리턴값:**
읽은 문자열을 캡슐화하는 String 오브젝트입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*Reset* - 재설정

**인터페이스:**

```
void Reset();
```

메시지 본문을 읽기 전용 모드로 전환하고 커서를 바이트 메시지 스트림의 시작 지점으로 옮깁니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotReadableException

*WriteBoolean* - 부울 값 쓰기

**인터페이스:**

```
void  WriteBoolean(Boolean value);
```

부울 값을 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 부울 값입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteByte* - 바이트 쓰기

**인터페이스:**

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

바이트를 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 바이트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteBytes* - 바이트 쓰기

**인터페이스:**

```
void  WriteBytes(Byte[] value);
```

바이트 배열을 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 바이트 배열입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteBytes* - 부분 바이트 배열 쓰기

**인터페이스:**

```
void  WriteBytes(Byte[] value, int offset, int length);
```

지정된 길이로 정의된 부분 바이트 배열을 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 바이트 배열입니다.

**offset (input)**
기록할 바이트 배열의 시작 지점입니다.

**length (input)**
기록할 바이트 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteChar* - 문자 쓰기

**인터페이스:**

```
void  WriteChar(Char value);
```

문자를 2바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 문자입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteDouble* - 배 정밀도 부동 소수점 수 쓰기

**인터페이스:**

```
void  WriteDouble(Double value);
```

배 정밀도 부동 소수점 수를 long 정수로 변환하고 long 정수를 8바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 배 정밀도 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteFloat* - 부동 소수점 수 쓰기

**인터페이스:**

```
void  WriteFloat(Single value);
```

부동 소수점 수를 정수로 변환하고 정수를 4바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteInt* - 정수 쓰기

**인터페이스:**

```
void  WriteInt(Int32 value);
```

정수를 4바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteLong - Long* 정수 쓰기

**인터페이스:**

```
void  WriteLong(Int64 value);
```

long 정수를 8바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 Long 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteObject* - 오브젝트 쓰기

**인터페이스:**

```
void  WriteObject(Object value);
```

지정된 오브젝트를 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 오브젝트이며, 원시 유형에 대한 참조여야 합니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteShort - Short* 정수 쓰기

**인터페이스:**

```
void  WriteShort(Int16 value);
```

short 정수를 2바이트(상위 바이트 우선)로 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 Short 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteUTF - UTF* 문자열 쓰기

**인터페이스:**

```
void  WriteUTF(String value);
```

UTF-8로 인코딩된 문자열을 바이트 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 문자열을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

### 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

clearBody, clearProperties, PropertyExists

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IConnection

Connection 오브젝트는 메시징 서버에 대한 애플리케이션의 활성 연결을 나타냅니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnection
```

Connection 오브젝트의 XMS 정의 특성 목록은 168 페이지의 『Connection 특성』의 내용를 참조하십시오.

### *.NETproperties*

*ClientID* - 클라이언트 *ID* 가져오기 및 설정

**인터페이스:**

```
String ClientID
      {
        get;
        set;
      }
```

연결에 대한 클라이언트 ID를 가져오고 설정합니다.

클라이언트 ID는 관리자가 ConnectionFactory에 미리 정의하거나 ClientID를 설정하여 지정할 수 있습니다.

클라이언트 ID는 발행/구독 도메인에서 지속 가능한 구독을 지원하는 데만 사용되며 포인트-투-포인트 도메인에서는 무시됩니다.

애플리케이션이 연결에 대한 클라이언트 ID를 설정할 경우 애플리케이션은 연결이 작성된 후 연결에서 다른 작업을 수행하기 전에 바로 클라이언트 ID를 설정해야 합니다. 이 시점 이후에 애플리케이션에서 클라이언트 ID를 설정하려고 하면 호출에서 IllegalStateException 예외가 발생합니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

**예외:**

- XMSException
- IllegalStateException
- InvalidClientIDException

*ExceptionListener* - 예외 리스너 가져오기 및 설정

**인터페이스:**

```
ExceptionListener ExceptionListener
    {
      get;
      set;
    }
```

연결에 등록된 예외 리스너를 가져오고 연결에 예외 리스너를 등록합니다.

연결에 예외 리스너가 등록되어 있지 않을 경우 이 메소드가 널을 리턴합니다. 예외 리스너가 이미 연결에 등록되어 있을 경우 예외 리스너 대신 널을 지정하여 등록을 취소할 수 있습니다.

예외 리스너의 사용에 대한 자세한 정보는 44 페이지의 『.NET의 메시지 및 예외 리스너』의 내용을 참조하십시오.

**예외:**

- XMSException

*Metadata* - 메타데이터 가져오기

**인터페이스:**

```
IConnectionMetaData MetaData
    {
      get;
    }
```

연결에 대한 메타데이터를 가져옵니다.

**예외:**

- XMSException

## 방법

*Close* - 연결 닫기

**인터페이스:**

```
void  Close();
```

연결을 닫습니다.

애플리케이션이 이미 닫힌 연결을 닫으려는 경우 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*CreateSession* - 세션 작성

**인터페이스:**

```
ISession CreateSession(Boolean transacted,
                       AcknowledgeMode acknowledgeMode);
```

세션을 작성합니다.

**매개변수:**

**transacted (input)**
`True` 값을 설정하면 세션이 트랜잭션되며, `False` 값을 설정하면 세션이 트랜잭션되지 않습니다.

브로커에 대한 실시간 연결의 경우 값은 `False`여야 합니다.

**acknowledgeMode (input)**
애플리케이션이 수신한 메시지를 수신확인하는 방법을 표시합니다. AcknowledgeMode 열거자에 있는 다음 값 중 하나를 사용해야 합니다.

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

브로커에 대한 실시간 연결의 경우 값은 `AcknowledgeMode.AutoAcknowledge` 또는 `AcknowledgeMode.DupsOkAcknowledge`여야 합니다.

트랜잭션된 세션인 경우 이 매개변수는 무시됩니다. 수신확인 모드에 대한 자세한 정보는 24 페이지의 『메시지 수신확인』의 내용을 참조하십시오.

**리턴값:**
Session 오브젝트입니다.

**예외:**

- XMSException

*Start* - 연결 시작

**인터페이스:**

```
void  Start();
```

연결에 대한 수신 메시지 전달을 시작하거나 다시 시작합니다. 연결이 이미 시작된 경우에는 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*Stop* - 연결 중지

**인터페이스:**

```
void  Stop();
```

연결에 대한 수신 메시지 전달을 중지합니다. 연결이 이미 중지된 경우에는 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IConnectionFactory

애플리케이션에서 연결 팩토리를 사용하여 연결을 작성합니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionFactory
```

ConnectionFactory 오브젝트의 XMS 정의 특성 목록은 169 페이지의 『ConnectionFactory의 특성』의 내용을 참조하십시오.

**관련 개념**

ConnectionFactories 오브젝트 및 Connection 오브젝트
ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

WebSphere MQ 큐 관리자에 대한 보안 연결
XMS .NET 애플리케이션이 WebSphere MQ 큐 관리자에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

WebSphere 서비스 통합 버스 메시징 엔진에 대한 보안 연결
XMS 애플리케이션이 WebSphere 서비스 통합 버스 메시징 엔진에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

**관련 태스크**
관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**
관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

## 방법

*CreateConnection* - 연결 팩토리 작성*(*기본 사용자 *ID* 사용*)*

**인터페이스:**

```
IConnection CreateConnection();
```

기본 특성을 사용하여 연결 팩토리를 작성합니다.

WebSphere MQ에 연결하는 중이고 XMSC_USERID가 설정되지 않은 경우 큐 관리자는 기본적으로 로그온된 사용자의 사용자 ID를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 WebSphere MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다.

**매개변수:**
없음

**예외:**

- XMSException

*CreateConnection* - 연결 작성*(*지정된 사용자 *ID* 사용*)*

**인터페이스:**

```
IConnection CreateConnection(String userId, String password);
```

지정된 사용자 ID를 사용하여 연결을 작성합니다.

WebSphere MQ에 연결하는 중이고 XMSC_USERID가 설정되지 않은 경우 큐 관리자는 기본적으로 로그온된 사용자의 사용자 ID를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 WebSphere MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다.

연결은 중지됨 모드로 작성됩니다. 애플리케이션이 **Connection.start()**를 호출하기 전에는 메시지가 전달되지 않습니다.

**매개변수:**

**userID (input)**
애플리케이션 인증에 사용될 사용자 ID를 캡슐화하는 String 오브젝트입니다. 널을 제공하면 인증 없이 연결이 작성됩니다.

**password (input)**
애플리케이션 인증에 사용될 비밀번호를 캡슐화하는 String 오브젝트입니다. 널을 제공하면 인증 없이 연결이 작성됩니다.

**리턴값:**
Connection 오브젝트입니다.

**예외:**

- XMSException
- XMS_X_SECURITY_EXCEPTION

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IConnectionMetaData

ConnectionMetaData 오브젝트는 연결에 대한 정보를 제공합니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionMetaData
```

ConnectionMetaData 오브젝트의 XMS 정의 특성 목록은 174 페이지의 『ConnectionMetaData 특성』의 내용을 참조하십시오.

### *.NETproperties*

*JMSXPropertyNames - JMS* 정의 메시지 특성 가져오기

**인터페이스:**

```
System.Collections.IEnumerator JMSXPropertyNames
    {
      get;
    }
```

연결에서 지원되는 JMS 정의된 메시지 특성의 이라는 이름 목록을 리턴합니다.

브로커에 대한 실시간 연결에서는 JMS 정의 메시지 특성이 지원되지 않습니다.

**예외:**

- XMSException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IDestination

목적지는 애플리케이션이 메시지를 보내는 위치이거나 애플리케이션이 메시지를 수신하는 소스입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IDestination
```

Destination 오브젝트의 XMS 정의 특성 목록은 174 페이지의 『Destination 특성』의 내용을 참조하십시오.

**관련 개념**

ConnectionFactories 오브젝트 및 Connection 오브젝트
ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

목적지
XMS 애플리케이션은 Destination 오브젝트를 사용하여 전송 중인 메시지의 목적지와 수신 중인 메시지의 소스를 지정합니다.

목적지 와일드 카드
XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

토픽 URI(Uniform Resource Identifier)
토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

큐 URI(Uniform Resource Identifier)
큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

임시 목적지
XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

### *.NETproperties*

*Name* - 목적지 이름 가져오기

**인터페이스:**

```
String Name
{
  get;
}
```

목적지의 이름을 가져옵니다. 이 이름은 큐 이름이나 토픽 이름을 캡슐화하는 문자열입니다.

**예외:**

- XMSException

*TypeId* - 목적지 유형 가져오기

**인터페이스:**

```
DestinationType TypeId
{
  get;
}
```

목적지의 유형을 가져옵니다. 목적지 유형의 값은 다음 중 하나입니다.

```
DestinationType.Queue
DestinationType.Topic
```

**예외:**

- XMSException

## 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

# ExceptionListener

**상속 계층 구조:**
없음

애플리케이션이 예외 리스너를 사용하여 연결 문제점을 비동기적으로 알립니다.

애플리케이션이 메시지를 비동기적으로 이용할 때만 연결을 사용하고 그 외의 용도로는 사용하지 않는 경우 애플리케이션에서 연결과 관련된 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다. 다른 상황에서 예외 리스너를 사용하면 XMS에 대한 다음 동기 호출까지 기다리는 대신 연결 문제점을 즉시 알 수 있습니다.

## 위임

*ExceptionListener* - 예외 리스너

**인터페이스:**

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

애플리케이션에 연결 문제점을 알립니다.

이 위임을 구현하는 메소드는 연결에 등록할 수 있습니다.

예외 리스너의 사용에 대한 자세한 정보는 44 페이지의 『.NET의 메시지 및 예외 리스너』의 내용을 참조하십시오.

**매개변수:**

**exception (input)**
XMS에서 작성된 예외에 대한 포인터입니다.

**리턴값:**
Void

## IllegalStateException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.Exception
           |
           +----IBM.XMS.IllegalStateException
```

XMS는 애플리케이션이 잘못되거나 부적절한 시간에 메소드를 호출하는 경우 또는 XMS가 요청된 조작에 적합한 상태가 아닌 경우, 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## InitialContext

애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

**상속 계층 구조:**
없음

**관련 개념**

InitialContext 특성
InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식
관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

관리 대상 오브젝트의 검색
XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

InitialContext 오브젝트
애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

## *.NETproperties*

*Environment* - 환경 가져오기

**인터페이스:**

```
Hashtable Environment
{
     get;
}
```

환경을 가져옵니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

## 구성자

*InitialContext* - 초기 컨텍스트 작성

**인터페이스:**

```
InitialContext(Hashtable env);
```

InitialContext 오브젝트를 작성합니다.

**매개변수:**

관리 오브젝트 저장소에 대한 연결을 설정하는 데 필요한 정보를 Hashtable 환경의 구성자에 제공합니다.

**예외:**

- XMSException

## 방법

*AddToEnvironment* - 환경에 새 특성 추가

**인터페이스:**

```
Object AddToEnvironment(String propName, Object propVal);
```

환경에 새 특성을 추가합니다.

**매개변수:**

**propName (input)**
추가할 특성의 이름을 캡슐화하는 String 오브젝트입니다.

**propVal (input)**
추가할 특성의 값입니다.

**리턴값:**
이전 특성 값입니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*Close* - 해당 컨텍스트 닫기

**인터페이스:**

```
void Close()
```

해당 컨텍스트를 닫습니다.

**매개변수:**
없음

**리턴값:**
없음

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*Lookup* - 초기 컨텍스트에서 오브젝트 검색

**인터페이스:**

```
Object Lookup(String name);
```

관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

**매개변수:**

**name (input)**
검색할 관리 오브젝트의 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 간단한 이름 또는 복잡한 이름일 수 있습니다. 자세한 정보는 58 페이지의 『관리 대상 오브젝트의 검색』의 내용을 참조하십시오.

**리턴값:**
검색할 오브젝트의 유형에 따라 IConnectionFactory 또는 IDestination입니다. 함수가 디렉토리에 액세스할 수 있지만 필요한 오브젝트를 찾을 수 없는 경우 널이 리턴됩니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*RemoveFromEnvironment* - 환경에서 특성 제거

**인터페이스:**

```
Object RemoveFromEnvironment(String propName);
```

환경에서 특성을 제거합니다.

**매개변수:**

**propName (input)**
제거할 특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
제거된 오브젝트입니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

## InvalidClientIDException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidClientIDException
```

XMS 는 애플리케이션이 연결에 대한 클라이언트 ID를 설정하려고 시도하지만 클라이언트 ID가 유효하지 않거나 이미 사용 중인 경우 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## InvalidDestinationException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidDestinationException
```

XMS는 애플리케이션에서 유효하지 않은 목적지를 지정한 경우에 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## InvalidSelectorException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidSelectorException
```

XMS는 애플리케이션에서 구문이 유효하지 않은 메시지 선택자 표현식을 제공할 경우에 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## IMapMessage

맵 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 이름-값 쌍 세트로 구성된 메시지입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IMapMessage
```

애플리케이션이 이름-값 쌍의 값을 가져올 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 이 암시적 변환 형식에 대한 자세한 정보는 71 페이지의 『맵 메시지』의 내용을 참조하십시오.

**관련 참조**

맵 메시지

맵 메시지의 본문에는 이름-값 쌍 세트가 포함됩니다. 이 때 각 값에는 연관된 데이터 유형이 있습니다.

### *.NETproperties*

*MapNames* - 맵 이름 가져오기

**인터페이스:**

```
System.Collections.IEnumerator MapNames
{
```

```
  get;
}
```

맵 메시지 본문에서 이름 목록을 가져옵니다.

**예외:**

- XMSException

## 방법

*GetBoolean* - 부울 값 가져오기

**인터페이스:**

```
Boolean GetBoolean(String name);
```

맵 메시지 본문에서 이름으로 식별되는 부울 값을 가져옵니다.

**매개변수:**

**name (input)**
부울 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 부울 값입니다.

**예외:**

- XMSException

*GetByte* - 바이트 가져오기

**인터페이스:**

```
Byte    GetByte(String name);
    Int16   GetSignedByte(String name);
```

맵 메시지 본문에서 이름으로 식별되는 바이트를 가져옵니다.

**매개변수:**

**name (input)**
바이트를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 바이트입니다. 바이트에 대한 데이터 변환은 수행되지 않습니다.

**예외:**

- XMSException

*GetBytes* - 바이트 가져오기

**인터페이스:**

```
Byte[]  GetBytes(String name);
```

맵 메시지 본문에서 이름으로 식별되는 바이트 배열을 가져옵니다.

**매개변수:**

**name (input)**
바이트 배열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
배열의 바이트 수입니다.

**예외:**

- XMSException

*GetChar* - 문자 가져오기

**인터페이스:**

```
Char    GetChar(String name);
```

맵 메시지 본문에서 이름으로 식별되는 문자를 가져옵니다.

**매개변수:**

**name (입력)**
문자를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 문자입니다.

**예외:**

- XMSException

*GetDouble* - 배 정밀도 부동 소수점 수 가져오기

**인터페이스:**

```
Double  GetDouble(String name);
```

맵 메시지 본문에서 이름으로 식별되는 배 정밀도 부동 소수점 수를 가져옵니다.

**매개변수:**

**name (input)**
배 정밀도 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 배 정밀도 부동 소수점 수입니다.

**예외:**

- XMSException

*GetFloat* - 부동 소수점 수 가져오기

**인터페이스:**

```
Single  GetFloat(String name);
```

맵 메시지 본문에서 이름으로 식별되는 부동 소수점 수를 가져옵니다.

**매개변수:**

**name (input)**
부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 부동 소수점 수입니다.

**예외:**

- XMSException

*GetInt* - 정수 가져오기

**인터페이스:**

```
Int32   GetInt(String name);
```

맵 메시지 본문에서 이름으로 식별되는 정수를 가져옵니다.

**매개변수:**

**name (input)**
정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 정수입니다.

**예외:**

- XMSException

*GetLong* - *Long* 정수 가져오기

**인터페이스:**

```
Int64   GetLong(String name);
```

맵 메시지 본문에서 이름으로 식별되는 long 정수를 가져옵니다.

**매개변수:**

**name (input)**
long 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 long 정수입니다.

**예외:**

- XMSException

*GetObject* - 오브젝트 가져오기

**인터페이스:**

```
Object  GetObject(String name);
```

맵 메시지 본문에서 이름-값 쌍 값에 대한 참조를 가져옵니다. 이름-값 쌍은 이름으로 식별됩니다.

**매개변수:**

**name (input)**
이름-값 쌍의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
다음 오브젝트 유형 중 하나인 값입니다.

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

**예외:**
XMSException

*GetShort - Short* 정수 가져오기

**인터페이스:**

```
Int16   GetShort(String name);
```

맵 메시지 본문에서 이름으로 식별되는 short 정수를 가져옵니다.

**매개변수:**

**name (input)**
short 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 short 정수입니다.

**예외:**

- XMSException

*GetString* - 문자열 가져오기

**인터페이스:**

```
String  GetString(String name);
```

맵 메시지 본문에서 이름으로 식별되는 문자열을 가져옵니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 문자열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
맵 메시지 본문에서 검색된 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 문자열입니다.

**예외:**

- XMSException

*ItemExists* - 이름-값 쌍이 있는지 검사

**인터페이스:**

```
Boolean ItemExists(String name);
```

맵 메시지 본문에 지정된 이름의 이름-값 쌍이 있는지 검사합니다.

**매개변수:**

**name (input)**
이름-값 쌍의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

- `True` - 맵 메시지 본문에 지정된 이름의 이름-값 쌍이 있는 경우
- `False` - 맵 메시지 본문에 지정된 이름의 이름-값 쌍이 없는 경우

**예외:**

- XMSException

*SetBoolean* - 부울 값 설정

**인터페이스:**

```
void  SetBoolean(String name, Boolean value);
```

맵 메시지 본문에서 부울 값을 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 부울 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 부울 값입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetByte* - 바이트 설정

**인터페이스:**

```
void  SetByte(String name, Byte value);
void  SetSignedByte(String name, Int16 value);
```

맵 메시지 본문에서 바이트를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 바이트를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 바이트입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetBytes* - 바이트 설정

**인터페이스:**

```
void  SetBytes(String name, Byte[] value);
```

맵 메시지 본문에서 바이트 배열을 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 바이트 배열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 바이트 배열입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetChar* - 문자 설정

**인터페이스:**

```
void  SetChar(String name, Char value);
```

맵 메시지 본문에서 2바이트 문자를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 문자를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 문자입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetDouble* - 배 정밀도 부동 소수점 수 설정

**인터페이스:**

```
void  SetDouble(String name, Double value);
```

맵 메시지 본문에서 배 정밀도 부동 소수점 수를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 배 정밀도 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 배 정밀도 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetFloat* - 부동 소수점 수 설정

**인터페이스:**

```
void  SetFloat(String name, Single value);
```

맵 메시지 본문에서 부동 소수점 수를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetInt* - 정수 설정

**인터페이스:**

```
void  SetInt(String name, Int32 value);
```

맵 메시지 본문에서 정수를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetLong - Long* 정수 설정

**인터페이스:**

```
void  SetLong(String name, Int64 value);
```

맵 메시지 본문에서 long 정수를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 long 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 long 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetObject* - 오브젝트 설정

**인터페이스:**

```
void  SetObject(String name, Object value);
```

맵 메시지 본문에서 값을 XMS 원시 유형으로 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 값을 포함하는 바이트 배열입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetShort - Short* 정수 설정

**인터페이스:**

```
void  SetShort(String name, Int16 value);
```

맵 메시지 본문에서 short 정수를 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 short 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 short 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*SetString* - 문자열 설정

**인터페이스:**

```
void  SetString(String name, String value);
```

맵 메시지 본문에서 문자열을 설정합니다.

**매개변수:**

**name (input)**
맵 메시지 본문에서 문자열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
설정할 문자열을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException

## 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

clearBody, clearProperties, PropertyExists

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

# IMessage

Message 오브젝트는 애플리케이션이 송신하거나 수신하는 메시지를 나타냅니다. IMessage는 IMapMessage와 같은 메시지 클래스의 수퍼클래스입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
```

Message 오브젝트의 JMS 메시지 헤더 필드 목록은 64 페이지의 『XMS 메시지의 헤더 필드』의 내용을 참조하십시오. Message 오브젝트의 JMS 정의 특성 목록은 66 페이지의 『메시지의 JMS 정의 특성』의 내용을 참조하십시오. Message 오브젝트의 IBM 정의 특성 목록은 66 페이지의 『IBM 정의 메시지 특성』의 내용을 참조하십시오. Message 오브젝트의 JMS_IBM_MQMD* 특성 목록은 178 페이지의 『JMS_IBM_MQMD* 특성』의 내용을 참조하십시오

메시지는 가비지 콜렉터에 의해 삭제됩니다. 메시지가 삭제되면 해당 메시지가 사용하고 있던 자원이 해제됩니다.

**관련 참조**
XMS 메시지
이 장에서는 XMS 메시지의 구조와 컨텐츠에 대해 설명하고 애플리케이션에서 XMS 메시지를 처리하는 방법을 제공합니다.

## *.NETproperties*

*GetJMSCorrelationID - JMSCorrelationID* 가져오기 및 설정

**인터페이스:**

```
String JMSCorrelationID
{
  get;
  set;
}
```

메시지의 상관 ID를 가져와서 String 오브젝트로 설정합니다.

**예외:**

- XMSException

*JMSDeliveryMode - JMSDeliveryMode* 가져오기 및 설정

**인터페이스:**

```
DeliveryMode JMSDeliveryMode
{
  get;
  set;
}
```

메시지 전달 모드를 가져오고 설정합니다.

메시지의 전달 모드는 다음 값 중 하나입니다.

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

전송되지 않은 새로 작성된 메시지의 경우 전달 모드는 `DeliveryMode.Persistent`지만, 브로커에 대한 실시간 연결의 경우 전달 모드는 `DeliveryMode.NonPersistent`입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSDeliveryMode를 설정하여 전달 모드를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 전달 모드를 리턴합니다.

**예외:**

- XMSException

*JMSDestination - JMSDestination* 가져오기 및 설정

**인터페이스:**

```
IDestination JMSDestination
{
  get;
  set;
}
```

메시지 목적지를 가져오고 설정합니다.

목적지는 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. JMSDestination의 값은 무시됩니다. 그러나 JMSDestination을 사용하여 수신된 메시지의 목적지를 변경할 수 있습니다.

전송되지 않은 새로 작성된 메시지의 경우 이 메소드는 송신하는 애플리케이션이 JMSDestination을 설정하여 목적지를 설정하지 않는 한 널 Destination 오브젝트를 리턴합니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSDestination을 설정하여 목적지를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 목적지에 대한 Destination 오브젝트를 리턴합니다.

**예외:**

- XMSException

*JMSExpiration - JMSExpiration* 가져오기 및 설정

**인터페이스:**

```
Int64 JMSExpiration
{
  get;
  set;
}
```

메시지의 만기 시간을 가져오고 설정합니다.

만기 시간은 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. 만기 시간 값은 전송 애플리케이션이 지정한 TTL(Time to Live)을 메시지 전송 시간에 더한 값으로, 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 JMSExpiration을 설정하여 다른 만기 시간을 설정하지 않는 한 만기 시간은 0입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSExpiration을 설정하여 만기 시간을 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 만기 시간을 리턴합니다.

TTL(Time to Live)이 0일 경우 IMessageProducer.send() 호출은 만기 시간을 0으로 설정하여 메시지가 만기되지 않음을 나타냅니다.

XMS는 만기된 메시지를 버리고 이를 애플리케이션에 전달하지 않습니다.

**예외:**

- XMSException

*JMSMessageID - JMSMessageID* 가져오기 및 설정

**인터페이스:**

```
String JMSMessageID
{
  get;
  set;
}
```

메시지의 메시지 ID를 가져와서 메시지 ID를 캡슐화하는 String 오브젝트로 설정합니다.

메시지 ID는 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSMessageID를 설정하여 메시지 ID를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 메시지 ID를 리턴합니다.

메시지에 메시지 ID가 없을 경우 이 메소드는 널을 리턴합니다.

**예외:**

- XMSException

*JMSPriority - JMSPriority* 가져오기 및 설정

**인터페이스:**

```
Int32 JMSPriority
{
  get;
  set;
}
```

메시지 우선순위를 가져오고 설정합니다.

우선순위는 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 JMSPriority를 설정하여 다른 우선순위를 설정하지 않는 한 우선순위는 4입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSPriority를 설정하여 우선순위를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 우선순위를 리턴합니다.

**예외:**

- XMSException

*JMSRedelivered - JMSRedelivered* 가져오기 및 설정

**인터페이스:**

```
Boolean JMSRedelivered
{
  get;
  set;
}
```

메시지가 다시 전달되는지 여부에 대한 표시를 가져와서 메시지가 다시 전달되는지 여부를 표시합니다. 이 표시는 메시지를 수신할 때 IMessageConsumer.receive() 호출에 의해 설정됩니다.

이 특성의 값은 다음과 같습니다.

- `True` - 메시지가 다시 전달되는 경우
- `False` - 메시지가 다시 전달되지 않는 경우

브로커에 대한 실시간 연결의 경우 이 값은 항상 `False`입니다.

메시지가 전송되기 전에 JMSRedelivered에 의해 설정된 다시 전달 표시는 메시지가 전송될 때 IMessageProducer.send() 호출에 의해 무시되고 메시지가 수신될 때 IMessageConsumer.receive() 호출에 의해 무시되고 대체됩니다. 그러나 JMSRedelivered를 사용하여 수신된 메시지에 대한 표시를 변경할 수 있습니다.

**예외:**

- XMSException

*JMSReplyTo - JMSReplyTo* 가져오기 및 설정

**인터페이스:**

```
IDestination JMSReplyTo
{
  get;
```

```
  set;
}
```

메시지에 대한 응답을 송신할 목적지를 가져오고 설정합니다.

이 특성의 값은 메시지에 대한 응답을 송신할 목적지에 대한 Destination 오브젝트입니다. 널 Destination 오브젝트는 응답이 예상되지 않음을 의미합니다.

**예외:**

- XMSException

*JMSTimestamp - JMSTimestamp* 가져오기 및 설정

**인터페이스:**

```
Int64 JMSTimestamp
{
  get;
  set;
}
```

메시지가 전송된 시간을 가져오고 설정합니다.

시간 소인은 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정되며, 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 JMSTimestamp를 설정하여 다른 시간 소인을 설정하지 않는 한 시간 소인은 0입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSTimestamp를 설정하여 시간 소인을 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 시간 소인을 리턴합니다.

**예외:**

- XMSException

**참고사항:**

1. 시간 소인이 정의되어 있지 않은 경우 이 메소드는 0만 리턴하고 예외가 발생되지는 않습니다.

*JMSType - JMSType* 가져오기 및 설정

**인터페이스:**

```
String JMSType
{
  get;
  set;
}
```

메시지 유형을 가져오고 설정합니다.

JMSType의 값은 메시지 유형을 캡슐화하는 문자열입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 유형입니다.

**예외:**

- XMSException

*PropertyNames* - 특성 가져오기

**인터페이스:**

```
System.Collections.IEnumerator PropertyNames
{
  get;
}
```

메시지의 이름 특성 목록을 가져옵니다.

**예외:**

- XMSException

## 방법

*Acknowledge* - 수신확인

**인터페이스:**

```
void  Acknowledge();
```

이 메시지를 비롯하여 세션이 이전에 수신한 메시지 중 수신확인되지 모든 메시지를 수신확인합니다.

세션의 수신확인 모드가 AcknowledgeMode.ClientAcknowledge인 경우 애플리케이션은 이 메소드를 호출할 수 있습니다. 세션에 다른 수신확인 모드가 있거나 트랜잭트된 경우 해당 메소드에 대한 호출이 무시됩니다.

수신되었지만 수신확인되지 않은 메시지는 다시 전달될 수 있습니다.

메시지 수신확인에 대한 자세한 정보는 24 페이지의 『메시지 수신확인』의 내용을 참조하십시오.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- IllegalStateException

*ClearBody* - 본문 지우기

**인터페이스:**

```
void  ClearBody();
```

메시지 본문을 지웁니다. 헤더 필드와 메시지 특성은 지워지지 않습니다.

애플리케이션이 메시지 본문을 지우면 본문은 새로 작성된 메시지의 빈 본문과 동일한 상태로 남아 있습니다. 새로 작성한 메시지의 빈 본문의 상태는 메시지 본문의 유형에 따라 다릅니다. 추가 정보는 68 페이지의 『XMS 메시지 본문』의 내용을 참조하십시오.

애플리케이션은 본문 상태에 관계없이 언제든지 메시지 본문을 지울 수 있습니다. 메시지 본문이 읽기 전용일 경우 애플리케이션이 본문에 기록할 수 있는 유일한 방법은 애플리케이션이 먼저 본문을 지우는 것입니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*ClearProperties* - 특성 지우기

**인터페이스:**

```
void  ClearProperties();
```

메시지 특성을 지웁니다. 헤더 필드와 메시지 본문은 지워지지 않습니다.

애플리케이션이 메시지 특성을 지울 경우 특성은 읽기 및 쓰기가 가능해집니다.

애플리케이션은 특성의 상태에 관계없이 언제든지 메시지 특성을 지울 수 있습니다. 메시지 특성이 읽기 전용일 경우 쓰기 가능한 특성이 될 수 있는 유일한 방법은 애플리케이션이 먼저 특성을 지우는 것입니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*PropertyExists* - 특성 존재 여부 검사

**인터페이스:**

```
Boolean PropertyExists(String propertyName);
```

지정된 이름의 특성이 메시지에 있는지 검사합니다.

**매개변수:**

**propertyName (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

- `True` - 지정된 이름의 특성이 메시지에 있는 경우
- `False` - 지정된 이름의 특성이 메시지에 없는 경우

**예외:**

- XMSException

## 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IMessageConsumer

애플리케이션에서 메시지 이용자를 사용하여 목적지로 송신된 메시지를 수신합니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageConsumer
```

MessageConsumer 오브젝트의 XMS 정의 특성 목록은 182 페이지의 『MessageConsumer 특성』의 내용을 참조하십시오.

### *.NETproperties*

*MessageListener* - 메시지 리스너 가져오기 및 설정

**인터페이스:**

```
MessageListener MessageListener
{
  get;
  set;
}
```

메시지 이용자에 등록된 메시지 리스너를 가져오고 메시지 이용자에 메시지 리스너를 등록합니다.

메시지 이용자에 메시지 리스너가 등록되지 않은 경우 MessageListener는 널입니다. 메시지 리스너가 이미 메시지 이용자에 등록된 경우 대신 널을 지정하여 등록을 취소할 수 있습니다.

메시지 리스너의 사용에 대한 자세한 정보는 44 페이지의 『.NET의 메시지 및 예외 리스너』의 내용을 참조하십시오.

**예외:**

- XMSException

*MessageSelector* - 메시지 선택자 가져오기

**인터페이스:**

```
String MessageSelector
{
  get;
}
```

메시지 이용자에 대한 메시지 선택자를 가져옵니다. 리턴값은 메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 메시지 선택자 표현식입니다. 메시지 이용자에 메시지 선택자가 없는 경우 MessageSelector의 값은 널 String 오브젝트입니다.

**예외:**

- XMSException

### 방법

*Close* - 메시지 이용자 닫기

**인터페이스:**

```
void  Close();
```

메시지 이용자를 닫습니다.

애플리케이션이 이미 닫힌 메시지 이용자를 닫으려는 경우 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*Receive* - 수신

**인터페이스:**

```
IMessage Receive();
```

메시지 이용자에 대한 다음 메시지를 수신합니다. 호출이 무기한으로 메시지를 기다리거나 메시지 이용자가 닫힐 때까지 기다립니다.

**매개변수:**
없음

**리턴값:**
Message 오브젝트에 대한 포인터입니다. 호출이 메시지를 기다리는 동안 메시지 이용자가 닫힌 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴합니다.

**예외:**

- XMSException

*Receive* - 수신*(*대기 간격 있음*)*

**인터페이스:**

```
IMessage Receive(Int64 delay);
```

메시지 이용자에 대한 다음 메시지를 수신합니다. 호출이 지정된 기간 동안만 메시지를 기다리거나 메시지 이용자가 닫힐 때까지 기다립니다.

**매개변수:**

**delay (input)**
호출이 메시지를 기다리는 시간(밀리초)입니다. 대기 간격을 0으로 지정하면 호출이 메시지를 무기한으로 기다립니다.

**리턴값:**
Message 오브젝트에 대한 포인터입니다. 대기 간격 중 메시지가 도착하지 않거나 호출이 메시지를 기다리는 동안 메시지 이용자가 닫힌 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴하지만 예외는 발생하지 않습니다.

**예외:**

- XMSException

*ReceiveNoWait* - 수신(대기 시간 없음)

**인터페이스:**

```
IMessage ReceiveNoWait();
```

메시지를 즉시 사용할 수 있는 경우 메시지 이용자의 다음 메시지를 수신합니다.

**매개변수:**
없음

**리턴값:**
Message 오브젝트에 대한 포인터입니다. 즉시 사용할 수 있는 메시지가 없는 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴합니다.

**예외:**

- XMSException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## MessageEOFException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageEOFException
```

XMS 는 애플리케이션이 바이트 메시지 본문을 읽을 때 XMS 가 바이트 메시지 스트림의 끝을 발견하는 경우 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## MessageFormatException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageFormatException
```

XMS XMS 이 (가) 유효하지 않은 형식의 메시지가 발견되면 이 예외가 발생합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## IMessageListener(위임)

**상속 계층 구조:**
없음

애플리케이션이 메시지 리스너를 사용하여 메시지를 비동기적으로 수신합니다.

## 위임

*MessageListener* - 메시지 리스너

**인터페이스:**

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

메시지를 메시지 이용자에게 비동기적으로 전달합니다.

이 위임을 구현하는 메소드는 연결에 등록할 수 있습니다.

메시지 리스너의 사용에 대한 자세한 정보는 44 페이지의 『.NET의 메시지 및 예외 리스너』의 내용을 참조하십시오.

**매개변수:**

**mesg (input)**
Message 오브젝트입니다.

**리턴값:**
Void

## MessageNotReadableException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.MessageNotReadableException
```

애플리케이션에서 쓰기 전용인 메시지 본문을 읽으려고 하는 경우 XMS에서 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## MessageNotWritableException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.MessageNotWritableException
```

XMS 는 애플리케이션이 읽기 전용인 메시지 본문에 쓰려고 시도하는 경우 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## IMessageProducer

애플리케이션은 메시지 작성자를 사용하여 메시지를 목적지에 송신합니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageProducer
```

MessageProducer 오브젝트의 XMS 정의 특성 목록은 182 페이지의 『MessageProducer 특성』의 내용을 참조하십시오.

### *.NETproperties*

*DeliveryMode* - 기본 전달 모드 가져오기 및 설정

**인터페이스:**

```
DeliveryMode DeliveryMode
{
  get;
  set;
}
```

메시지 작성자가 전송한 메시지의 기본 전달 모드를 가져오고 설정합니다.

기본 전달 모드는 다음 값 중 하나입니다.

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

브로커에 대한 실시간 연결의 경우 값은 `DeliveryMode.NonPersistent`여야 합니다.

기본값은 `DeliveryMode.Persistent`지만, 브로커에 대한 실시간 연결의 기본값은 `DeliveryMode.NonPersistent`입니다.

**예외:**

- XMSException

*Destination* - 목적지 가져오기

**인터페이스:**

```
IDestination Destination
{
  get;
}
```

메시지 작성자에 대한 목적지를 가져옵니다.

**매개변수:**
없음

**리턴값:**
Destination 오브젝트입니다. 메시지 작성자에 목적지가 없으면 이 메소드는 널 Destination 오브젝트를 리턴합니다.

**예외:**

- XMSException

*DisableMsgID* - 메시지 *ID* 사용 안함 플래그 가져오기 및 설정

**인터페이스:**

```
Boolean DisableMessageID
{
  get;
  set;
}
```

수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시켜야 하는지 여부를 나타내는 표시를 가져오고, 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시킬지 여부를 표시합니다.

큐 관리자에 대한 연결이나 브로커에 대한 실시간 연결에서는 이 플래그가 무시되고, 서비스 통합 버스에 대한 연결에서는 이 플래그가 적용됩니다.

DisabledMsgID의 값은 다음과 같습니다.

- `True` - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시킬 필요가 없는 경우
- `False` - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시켜야 하는 경우

**예외:**

- XMSException

*DisableMsgTS* - 시간소인 사용 안함 플래그 가져오기 및 설정

**인터페이스:**

```
Boolean DisableMessageTimestamp
{
  get;
  set;
}
```

수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시켜야 하는지 여부를 나타내는 표시를 가져오고, 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시킬지 여부를 표시합니다.

브로커에 대한 실시간 연결에서는 이 플래그가 무시되고, 큐 관리자에 대한 연결이나 서비스 통합 버스에 대한 연결에서는 이 플래그가 적용됩니다.

DisableMsgTS의 값은 다음과 같습니다.

- `True` - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시킬 필요가 없는 경우
- `False` - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시켜야 하는 경우

**리턴값:**

**예외:**

- XMSException

*Priority* - 기본 우선순위 가져오기 및 설정

**인터페이스:**

```
Int32 Priority
```

```
{
  get;
  set;
}
```

메시지 작성자가 보낸 메시지의 기본 우선순위를 가져오고 설정합니다.

기본 메시지 우선순위 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다.

브로커에 대한 실시간 연결에서는 메시지의 우선순위가 무시됩니다.

**예외:**

- XMSException

*TimeToLive* - 기본 *TTL(Time to Live)* 가져오기 및 설정

**인터페이스:**

```
Int64 TimeToLive
{
  get;
  set;
}
```

만료될 때까지의 메시지가 존재하는 기본 시간을 가져오고 설정합니다.

이 시간은 메시지 작성자가 메시지를 보내는 시간부터 측정되며 밀리초 단위의 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다.

브로커에 대한 실시간 연결의 경우 이 값은 항상 0입니다.

**예외:**

- XMSException

## 방법

*Close* - 메시지 작성자 닫기

**인터페이스:**

```
void  Close();
```

메시지 작성자를 닫습니다.

애플리케이션이 이미 닫힌 메시지 작성자를 닫으려는 경우 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*Send* - 송신

**인터페이스:**

```
void Send(IMessage msg) ;
```

메시지 생성자가 작성될 때 지정된 대상으로 메시지를 전송합니다. 메시지 작성자 기본 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

**매개변수:**

**msg (input)**
Message 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageFormatException
- InvalidDestinationException

*Send* - 송신*(*전달 모드*,* 우선순위 및 *TTL(Time to Live)* 지정*)*

**인터페이스:**

```
void Send(IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive);
```

메시지 생성자가 작성될 때 지정된 대상으로 메시지를 전송합니다. 지정된 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

**매개변수:**

**msg (input)**
Message 오브젝트입니다.

**deliveryMode (input)**
메시지의 전달 모드이며, 다음 값 중 하나여야 합니다.

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

브로커에 대한 실시간 연결의 경우 값은 `DeliveryMode.NonPersistent`여야 합니다.

**priority (input)**
메시지의 우선순위입니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다. 브로커에 대한 실시간 연결에서는 이 값이 무시됩니다.

**timeToLive (input)**
밀리초 단위의 메시지 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다. 브로커에 대한 실시간 연결의 경우 이 값은 0이어야 합니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageFormatException
- InvalidDestinationException

- IllegalStateException

*Send* - 송신*(*지정된 목적지로 보내기*)*

**인터페이스:**

```
void Send(IDestination dest, IMessage msg) ;
```

메시지 생성자가 작성될 때 대상이 지정되지 않은 메시지 생성자를 사용하는 경우 지정된 대상으로 메시지를 전송합니다. 메시지 작성자 기본 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

일반적으로 메시지 작성자를 생성할 때 목적지를 지정합니다. 그렇지 않을 경우 메시지를 전송할 때마다 목적지를 지정해야 합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

**msg (input)**
Message 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageFormatException
- InvalidDestinationException

*Send* - 송신*(*지정된 목적지로 보내기, 전달 모드, 우선순위 및 *TTL(Time to Live)* 지정*)*

**인터페이스:**

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

메시지 생성자가 작성될 때 대상이 지정되지 않은 메시지 생성자를 사용하는 경우 지정된 대상으로 메시지를 전송합니다. 지정된 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

일반적으로 메시지 작성자를 생성할 때 목적지를 지정합니다. 그렇지 않을 경우 메시지를 전송할 때마다 목적지를 지정해야 합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

**msg (input)**
Message 오브젝트입니다.

**deliveryMode (input)**
메시지의 전달 모드이며, 다음 값 중 하나여야 합니다.

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

브로커에 대한 실시간 연결의 경우 값은 `DeliveryMode.NonPersistent`여야 합니다.

**priority (input)**
메시지의 우선순위입니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다. 브로커에 대한 실시간 연결에서는 이 값이 무시됩니다.

**timeToLive (input)**
밀리초 단위의 메시지 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다. 브로커에 대한 실시간 연결의 경우 이 값은 0이어야 합니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IObjectMessage

오브젝트 메시지는 본문이 직렬화된 Java 또는 .NET 오브젝트로 구성된 메시지입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IObjectMessage
```

**관련 참조**
오브젝트 메시지
오브젝트 메시지의 본문에는 직렬화된 Java 또는 .NET 오브젝트가 있습니다.

### *.NETproperties*

*Object* - 오브젝트를 바이트로 가져오기 및 설정

**인터페이스:**

```
System.Object Object
    {
      get;
      set;
    }

    Byte[] GetObject();
```

오브젝트 메시지의 본문을 구성하는 오브젝트를 가져오고 설정합니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException
- MessageNotWritableException

### 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

clearBody, clearProperties, PropertyExists

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IPropertyContext

IPropertyContext는 특성을 가져오고 설정하는 메소드를 포함하는 추상 수퍼클래스입니다. 이러한 메소드는 다른 클래스에서 상속합니다.

**상속 계층 구조:**
없음

### 방법

*GetBooleanProperty* - 부울 특성 가져오기

**인터페이스:**

```
Boolean GetBooleanProperty(String property_name);
```

지정된 이름의 부울 특성 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetByteProperty* - 바이트 특성 가져오기

**인터페이스:**

```
Byte    GetByteProperty(String property_name) ;
Int16   GetSignedByteProperty(String property_name) ;
```

이름으로 식별되는 바이트 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetBytesProperty* - 바이트 배열 특성 가져오기

**인터페이스:**

```
Byte[]  GetBytesProperty(String property_name) ;
```

이름으로 식별되는 바이트 배열 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
배열의 바이트 수입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetCharProperty* - 문자 특성 가져오기

**인터페이스:**

```
Char    GetCharProperty(String property_name) ;
```

이름으로 식별되는 2바이트 문자 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetDoubleProperty* - 배 정밀도 부동 소수점 특성 가져오기

**인터페이스:**

```
Double  GetDoubleProperty(String property_name) ;
```

이름으로 식별되는 배 정밀도 부동 소수점 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetFloatProperty* - 부동 소수점 특성 가져오기

**인터페이스:**

```
Single  GetFloatProperty(String property_name) ;
```

이름으로 식별되는 부동 소수점 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetIntProperty - GetIntProperty*

**인터페이스:**

```
Int32   GetIntProperty(String property_name) ;
```

이름으로 식별되는 정수 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetLongProperty - Long* 정수 특성 가져오기

**인터페이스:**

```
Int64   GetLongProperty(String property_name) ;
```

이름으로 식별되는 Long 정수 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetObjectProperty* - 오브젝트 특성 가져오기

**인터페이스:**

```
Object  GetObjectProperty( String property_name) ;
```

이름으로 식별되는 특성의 값과 데이터 유형을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값이며, 다음 오브젝트 유형 중 하나입니다.

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetShortProperty - Short* 정수 특성 가져오기

**인터페이스:**

```
Int16   GetShortProperty(String property_name) ;
```

이름으로 식별되는 Short 정수 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성의 값입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*GetStringProperty - GetStringProperty*

**인터페이스:**

```
String  GetStringProperty(String property_name) ;
```

이름으로 식별되는 문자열 특성의 값을 가져옵니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
특성 값 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 문자열입니다.

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException

*SetBooleanProperty* - 부울 특성 설정

**인터페이스:**

```
void SetBooleanProperty( String property_name,  Boolean value) ;
```

이름으로 식별되는 부울 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetByteProperty* - 바이트 특성 설정

**인터페이스:**

```
void SetByteProperty( String property_name,  Byte value) ;
      void SetSignedByteProperty( String property_name,  Int16 value) ;
```

이름으로 식별되는 바이트 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetBytesProperty* - 바이트 배열 특성 설정

**인터페이스:**

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

이름으로 식별되는 바이트 배열 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
바이트 배열을 나타내는 특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetCharProperty* - 문자 특성 설정

**인터페이스:**

```
void SetCharProperty( String property_name,  Char value) ;
```

이름으로 식별되는 2바이트 문자 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetDoubleProperty* - 배 정밀도 부동 소수점 특성 설정

**인터페이스:**

```
void SetDoubleProperty( String property_name,  Double value) ;
```

이름으로 식별되는 배 정밀도 부동 소수점 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetFloatProperty* - 부동 소수점 특성 설정

**인터페이스:**

```
void SetFloatProperty( String property_name,  Single value) ;
```

이름으로 식별되는 부동 소수점 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetIntProperty* - 정수 특성 설정

**인터페이스:**

```
void SetIntProperty( String property_name,  Int32 value) ;
```

이름으로 식별되는 정수 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetLongProperty* - *Long* 정수 특성 설정

**인터페이스:**

```
void SetLongProperty( String property_name,  Int64 value) ;
```

이름으로 식별되는 long 정수 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetObjectProperty* - 오브젝트 특성 설정

**인터페이스:**

```
void SetObjectProperty( String property_name,  Object value) ;
```

이름으로 식별되는 특성의 값과 데이터 유형을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**objectType (input)**
특성의 값이며, 다음 오브젝트 유형 중 하나여야 합니다.

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

**value (input)**
바이트 배열을 나타내는 특성의 값입니다.

**length (input)**
배열의 바이트 수입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetShortProperty - Short* 정수 특성 설정

**인터페이스:**

```
void SetShortProperty( String property_name,  Int16 value) ;
```

이름으로 식별되는 short 정수 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성의 값입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

*SetStringProperty* - 문자열 특성 설정

**인터페이스:**

```
void SetStringProperty( String property_name,  String value);
```

이름으로 식별되는 문자열 특성의 값을 설정합니다.

**매개변수:**

**property_name (input)**
특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**
특성 값 문자열을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
Void

**스레드 컨텍스트:**
서브클래스에 의해 결정됨

**예외:**

- XMSException
- MessageNotWritableException

## IQueueBrowser

애플리케이션이 큐 브라우저를 사용하여 메시지를 제거하지 않고 큐에서 메시지를 찾습니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
   |
   +----IBM.XMS.IQueueBrowser
```

## *.NET* 특성

*MessageSelector* - 메시지 선택자 가져오기

**인터페이스:**

```
String MessageSelector
{
  get;
}
```

큐 브라우저에 대한 메시지 선택자를 가져옵니다.

메시지 선택자는 메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 메시지 선택자 표현식입니다. 큐 브라우저에 메시지 선택자가 없으면 이 메소드는 널 String 오브젝트를 리턴합니다.

**예외:**

- XMSException

*Queue* - 큐 가져오기

**인터페이스:**

```
IDestination Queue
{
  get;
}
```

큐를 나타내는 목적지 오브젝트로 큐 브라우저와 연관된 큐를 가져옵니다.

**예외:**

- XMSException

## 방법

*Close* - 큐 브라우저 닫기

**인터페이스:**

```
void  Close();
```

큐 브라우저를 닫습니다.

애플리케이션이 이미 닫힌 큐 브라우저를 닫으려는 경우 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException

*GetEnumerator* - 메시지 가져오기

**인터페이스:**

```
IEnumerator GetEnumerator();
```

큐에 있는 메시지 목록을 가져옵니다.

이 메소드는 Message 오브젝트의 목록을 캡슐화하는 열거자를 리턴합니다. Message 오브젝트의 순서는 큐에서 메시지를 검색하는 순서와 동일합니다. 애플리케이션은 열거자를 사용하여 각 메시지를 교대로 찾아볼 수 있습니다.

열거자는 메시지가 큐에 넣어지고 큐에서 제거될 때 동적으로 업데이트됩니다. 애플리케이션이 IEnumerator.MoveNext()를 호출하여 큐의 다음 메시지를 찾아볼 때마다 메시지는 큐의 현재 컨텐츠를 반영합니다.

애플리케이션이 이 메소드를 큐 브라우저에 두 번 이상 호출하는 경우 각 호출은 새 열거자를 리턴합니다. 그러므로 애플리케이션은 둘 이상의 열거자를 사용하여 큐의 메시지를 찾아보고 큐 내에 여러 위치를 유지보수할 수 있습니다.

**매개변수:**
없음

**리턴값:**
Iterator 오브젝트입니다.

**예외:**

- XMSException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## Requestor

애플리케이션이 요청자를 사용하여 요청 메시지를 보내고 잠시 대기한 후 응답을 수신합니다.

**상속 계층 구조:**
없음

### 구성자

*Requestor* - 요청자 작성

**인터페이스:**

```
Requestor(ISession sess, IDestination dest);
```

요청자를 작성합니다.

**매개변수:**

**sess (input)**
Session 오브젝트입니다. 세션을 트랜잭션하지 않아야 하고 다음 수신확인 모드 중 하나가 있어야 합니다.

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

**dest (input)**
애플리케이션이 요청 메시지를 보낼 수 있는 목적지를 나타내는 Destination 오브젝트입니다.

**스레드 컨텍스트:**
요청자와 연관된 세션입니다.

**예외:**

- XMSException

## 방법

*Close* - 요청자 닫기

**인터페이스:**

```
void  Close();
```

요청자를 닫습니다.

애플리케이션이 이미 닫힌 요청자를 닫으려는 경우 호출이 무시됩니다.

**참고:** 애플리케이션이 요청자를 닫아도 연관된 세션은 닫히지 않습니다. 이런 점에서 XMS는 JMS와 다르게 작동합니다.

**매개변수:**
없음

**리턴값:**
Void

**스레드 컨텍스트:**
임의

**예외:**

- XMSException

*Request* - 응답 요청

**인터페이스:**

```
IMessage Request(IMessage requestMessage);
```

요청 메시지를 전송한 후 기다렸다가 요청 메시지를 수신하는 애플리케이션에서 응답을 수신합니다.

응답을 수신하거나 세션이 끝날 때까지(둘 중에서 빠른 쪽이 적용됨) 이 메소드에 대한 호출은 차단됩니다.

**매개변수:**

**requestMessage (input)**
요청 메시지를 캡슐화하는 Message 오브젝트입니다.

**리턴값:**
응답 메시지를 캡슐화하는 Message 오브젝트에 대한 포인터입니다.

**스레드 컨텍스트:**
요청자와 연관된 세션입니다.

**예외:**

- XMSException

## ResourceAllocationException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.ResourceAllocationException
```

XMS가 메소드에서 요구하는 자원을 할당할 수 없는 경우 XMS는 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## SecurityException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.SecurityException
```

XMS는 애플리케이션 인증을 위해 제공한 사용자 ID와 비밀번호가 거부된 경우 이 예외를 처리합니다. XMS는 권한 검사에 실패하여 메소드가 완료되지 않은 경우에도 이 예외를 발생시킵니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## ISession

세션은 메시지 송신 및 수신을 위한 단일 스레드 컨텍스트입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.ISession
```

Session 오브젝트의 XMS 정의 특성 목록은 182 페이지의 『Session 특성』의 내용을 를 참조하십시오.

### *.NET* 특성

*AcknowledgeMode* - 수신확인 모드 가져오기

**인터페이스:**

```
AcknowledgeMode AcknowledgeMode
    {
      get;
    }
```

세션을 사용할 수신확인 모드를 가져오십시오.

세션을 작성할 때 수신확인 모드를 지정합니다.

세션이 트랜잭트되지 않은 경우, 수신확인 모드는 다음 값 중 하나입니다.

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

수신확인 모드에 대한 자세한 정보는 24 페이지의 『메시지 수신확인』의 내용을 참조하십시오.

트랜잭트된 세션에 수신확인 모드가 없습니다. 세션이 트랜잭션되면 메소드가 `AcknowledgeMode.SessionTransacted`를 대신 리턴합니다.

**예외:**

- XMSException

*Transacted* - 트랜잭션되었는지 여부를 판별

**인터페이스:**

```
Boolean Transacted
    {
      get;
    }
```

세션이 트랜잭션되었는지 여부를 판별합니다.

트랜잭션 상태는 다음과 같습니다.

- `True` - 세션이 트랜잭트된 경우
- `False` - 세션이 트랜잭트되지 않은 경우

브로커에 대한 실시간 연결의 경우, 이 메소드는 항상 `False`를 리턴합니다.

**예외:**

- XMSException

## 방법

*Close* - 세션 닫기

**인터페이스:**

```
void Close();
```

세션을 닫습니다. 세션이 트랜잭션되면 진행 중인 트랜잭션이 롤백됩니다.

애플리케이션이 이미 닫힌 세션을 닫으려는 경우 호출이 무시됩니다.

**매개변수:**
없음

**리턴값:**
Void

**스레드 컨텍스트:**
임의

**예외:**

- XMSException

*Commit* - 커미트

**인터페이스:**

```
void Commit();
```

현재 트랜잭션에서 처리된 모든 메시지를 커미트합니다.

세션은 트랜잭션된 세션이어야 합니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- IllegalStateException
- TransactionRolledBackException

*CreateBrowser* - 큐 브라우저 작성

**인터페이스:**

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

지정된 큐에 대한 큐 브라우저를 작성합니다.

**매개변수:**

**queue (input)**
큐를 나타내는 Destination 오브젝트입니다.

**리턴값:**
QueueBrowser 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException

*CreateBrowser* - 큐 브라우저 작성*(*메시지 선택자 사용*)*

**인터페이스:**

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

메시지 선택자를 사용하여 지정된 큐에 대한 큐 브라우저를 작성합니다.

**매개변수:**

**queue (input)**
큐를 나타내는 Destination 오브젝트입니다.

**selector (input)**
메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 큐 브라우저에 전달됩니다.

널 String 오브젝트는 큐 브라우저에 대한 메시지 선택자가 없음을 의미합니다.

**리턴값:**
QueueBrowser 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException
- InvalidSelectorException

*CreateBytesMessage* - 바이트 메시지 작성

**인터페이스:**

```
IBytesMessage CreateBytesMessage();
```

바이트 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
BytesMessage 오브젝트입니다.

**예외:**

- XMSException
- IllegalStateException(세션이 닫힘)

*CreateConsumer* - 이용자 작성

**인터페이스:**

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

지정된 목적지에 대한 메시지 이용자를 작성합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

**리턴값:**
MessageConsumer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException

*CreateConsumer* - 이용자 작성*(*메시지 선택자 사용*)*

**인터페이스:**

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector) ;
```

메시지 선택자를 사용하여 지정된 목적지에 대한 메시지 이용자를 작성합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

**selector (input)**
메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 메시지 이용자에 전달됩니다.

널 String 오브젝트는 메시지 이용자에 대한 메시지 선택자가 없음을 의미합니다.

**리턴값:**
MessageConsumer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException
- InvalidSelectorException

*CreateConsumer* - 이용자 작성*(*메시지 선택자 및 로컬 메시지 플래그 사용*)*

**인터페이스:**

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector,
                                Boolean noLocal) ;
```

메시지 선택자를 사용하여 지정된 목적지에 대한 메시지 이용자를 작성합니다. 이 때 목적지가 토픽인 경우 메시지 이용자가 자체 연결을 통해 공개된 메시지를 수신할지 여부도 지정합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

**selector (input)**
메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 메시지 이용자에 전달됩니다.

널 String 오브젝트는 메시지 이용자에 대한 메시지 선택자가 없음을 의미합니다.

**noLocal (input)**
값을 `True`로 설정하면 메시지 이용자가 자체 연결을 통해 발행된 메시지를 수신하지 않습니다. 값을 `False`로 설정하면 메시지 이용자가 자체 연결을 통해 발행된 메시지를 수신합니다. 기본값은 `False`입니다.

**리턴값:**
MessageConsumer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException
- InvalidSelectorException

*CreateDurableSubscriber* - 지속 가능한 구독자 작성

**인터페이스:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription) ;
```

지정된 토픽에 대해 지속 가능한 구독자를 작성합니다.

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

지속 가능한 구독자에 대한 자세한 정보는 31 페이지의 『지속 가능 구독자』의 내용을 참조하십시오.

**매개변수:**

**dest (input)**
토픽을 나타내는 Destination 오브젝트입니다. 토픽은 임시 토픽이 아니어야 합니다.

**subscription (input)**
지속 가능한 구독을 식별하는 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 연결에 대한 클라이언트 ID 내에서 고유해야 합니다.

**리턴값:**
지속 가능한 구독자를 나타내는 MessageConsumer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException

*CreateDurableSubscriber* - 지속 가능한 구독자 작성*(*메시지 선택자 및 로컬 메시지 플래그 사용*)*

**인터페이스:**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription,
                                         String selector,
                                         Boolean noLocal) ;
```

메시지 선택자를 사용하여 지정된 토픽에 대한 지속 가능한 구독자를 작성합니다. 이 때 지속 가능한 구독자가 자체 연결을 통해 공개된 메시지를 수신할지 여부도 지정합니다.

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

지속 가능한 구독자에 대한 자세한 정보는 31 페이지의 『지속 가능 구독자』의 내용을 참조하십시오.

**매개변수:**

**dest (input)**
토픽을 나타내는 Destination 오브젝트입니다. 토픽은 임시 토픽이 아니어야 합니다.

**subscription (input)**
지속 가능한 구독을 식별하는 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 연결에 대한 클라이언트 ID 내에서 고유해야 합니다.

**selector (input)**
메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 지속 가능한 구독자에 전달됩니다.

널 String 오브젝트는 지속 가능한 구독자에 대한 메시지 선택자가 없음을 의미합니다.

**noLocal (input)**
값을 `True`로 설정하면 지속 가능한 구독자가 자체 연결을 통해 발행된 메시지를 수신하지 않습니다. 값을 `False`로 설정하면 지속 가능한 구독자가 자체 연결을 통해 발행된 메시지를 수신하지 않습니다. 기본값은 `False`입니다.

**리턴값:**
지속 가능한 구독자를 나타내는 MessageConsumer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException

- InvalidSelectorException

*CreateMapMessage* - 맵 메시지 작성

**인터페이스:**

```
IMapMessage CreateMapMessage();
```

맵 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
MapMessage 오브젝트입니다.

**예외:**

- XMSException
- IllegalStateException(세션이 닫힘)

*CreateMessage* - 메시지 작성

**인터페이스:**

```
IMessage CreateMessage();
```

본문이 없는 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
Message 오브젝트입니다.

**예외:**

- XMSException
- IllegalStateException(세션이 닫힘)

*CreateObjectMessage* - 오브젝트 메시지 작성

**인터페이스:**

```
IObjectMessage CreateObjectMessage();
```

오브젝트 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
ObjectMessage 오브젝트입니다.

**예외:**

- XMSException
- IllegalStateException(세션이 닫힘)

*CreateProducer* - 작성자 작성

**인터페이스:**

```
IMessageProducer CreateProducer(IDestination dest) ;
```

메시지를 지정된 목적지로 전송하는 메시지 작성자를 작성합니다.

**매개변수:**

**dest (input)**
Destination 오브젝트입니다.

널 Destination 오브젝트를 지정하면 메시지 작성자가 목적지 없이 작성됩니다. 이 경우 애플리케이션은 메시지 작성자를 사용하여 메시지를 전송할 때마다 목적지를 지정해야 합니다.

**리턴값:**
MessageProducer 오브젝트입니다.

**예외:**

- XMSException
- InvalidDestinationException

*CreateQueue* - 큐 작성

**인터페이스:**

```
IDestination CreateQueue(String queue) ;
```

메시지 서버의 큐를 나타내는 Destination 오브젝트를 작성합니다.

이 메소드는 메시지 서버에 큐를 작성하지 않습니다. 따라서 애플리케이션이 이 메소드를 호출하려면 먼저 큐를 작성해야 합니다.

**매개변수:**

**queue (input)**
큐 이름을 캡슐화하거나 큐를 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**
큐를 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*CreateStreamMessage* - 스트림 메시지 작성

**인터페이스:**

```
IStreamMessage CreateStreamMessage();
```

스트림 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
StreamMessage 오브젝트입니다.

**예외:**

- XMSException
- XMS_ILLEGAL_STATE_EXCEPTION

*CreateTemporaryQueue* - 임시 큐 작성

**인터페이스:**

```
IDestination CreateTemporaryQueue() ;
```

임시 큐를 작성합니다.

임시 큐의 범위는 연결입니다. 연결에 의해 작성된 세션만 임시 큐를 사용할 수 있습니다.

임시 큐는 명시적으로 삭제되거나 연결이 끊길 때까지(둘 중에서 빠른 쪽이 적용됨) 그대로 유지됩니다.

임시 큐에 대한 자세한 정보는 30 페이지의 『임시 목적지』의 내용을 참조하십시오.

**매개변수:**
없음

**리턴값:**
임시 큐를 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*CreateTemporaryTopic* - 임시 토픽 작성

**인터페이스:**

```
IDestination CreateTemporaryTopic() ;
```

임시 토픽을 작성합니다.

임시 토픽의 범위는 연결입니다. 연결에 의해 작성된 세션만 임시 토픽을 사용할 수 있습니다.

임시 토픽은 명시적으로 삭제되거나 연결이 끊길 때까지(둘 중에서 빠른 쪽이 적용됨) 그대로 유지됩니다.

임시 토픽에 대한 자세한 정보는 30 페이지의 『임시 목적지』의 내용을 참조하십시오.

**매개변수:**
없음

**리턴값:**
임시 토픽을 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*CreateTextMessage* - 텍스트 메시지 작성

**인터페이스:**

```
ITextMessage CreateTextMessage();
```

본문이 비어 있는 텍스트 메시지를 작성합니다.

**매개변수:**
없음

**리턴값:**
TextMessage 오브젝트입니다.

**예외:**

- XMSException

*CreateTextMessage* - 텍스트 메시지 작성*(*초기화됨*)*

**인터페이스:**

```
ITextMessage CreateTextMessage(String initialValue);
```

지정된 텍스트로 본문이 초기화되는 텍스트 메시지를 작성합니다.

**매개변수:**

**initialValue (input)**
텍스트 메시지의 본문을 초기화하는 텍스트를 캡슐화하는 String 오브젝트입니다.

없음

**리턴값:**
TextMessage 오브젝트입니다.

**예외:**

- XMSException

*CreateTopic* - 토픽 작성

**인터페이스:**

```
IDestination CreateTopic(String topic) ;
```

토픽을 나타내는 Destination 오브젝트를 작성합니다.

**매개변수:**

**topic (input)**
토픽 이름을 캡슐화하거나 토픽을 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**
토픽을 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*Recover* - 복구

**인터페이스:**

```
void Recover();
```

세션을 복구합니다. 메시지 전달이 중지되었다가 수신확인되지 않은 가장 오래된 메시지를 사용하여 재시작됩니다.

세션은 트랜잭션된 세션이 아니어야 합니다.

세션 복구에 대한 자세한 정보는 24 페이지의 『메시지 수신확인』의 내용을 참조하십시오.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- IllegalStateException

*Rollback* - 롤백

**인터페이스:**

```
void Rollback();
```

현재 트랜잭션에서 처리된 모든 메시지를 커미트합니다.

세션은 트랜잭션된 세션이어야 합니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- IllegalStateException

*Unsubscribe* - 구독 해제

**인터페이스:**

```
void Unsubscribe(String subscription);
```

지속 가능한 구독을 삭제합니다. 메시징 서버는 유지보수 중인 지속 가능 구독의 레코드를 삭제하고 지속 가능 구독자에게 추가 메시지를 전송하지 않습니다.

다음과 같은 경우 애플리케이션은 지속 가능한 구독을 삭제할 수 없습니다.

- 지속 가능한 구독에 대해 활성 메시지 이용자가 있는 경우
- 이용된 메시지가 보류 중인 트랜잭션의 일부인 경우
- 이용된 메시지가 수신확인되지 않은 경우

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

**매개변수:**

**subscription (input)**
지속 가능한 구독을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException

- InvalidDestinationException
- IllegalStateException

### 상속된 특성 및 메소드

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IStreamMessage

스트림 메시지는 본문이 각 값에 연관된 데이터 유형이 있는 값 스트림으로 구성된 메시지입니다. 본문의 컨텐츠는 순차적으로 쓰여지고 읽힙니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IStreamMessage
```

애플리케이션이 메시지 스트림에서 값을 읽을 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 이 암시적 변환 형식에 대한 자세한 정보는 72 페이지의 『스트림 메시지』의 내용을 참조하십시오.

**관련 참조**

스트림 메시지
스트림 메시지의 본문은 값 스트림을 포함합니다. 이 때 각 값은 연관된 데이터 유형을 갖습니다.

### 방법

*ReadBoolean* - 부울 값 읽기

**인터페이스:**

```
Boolean ReadBoolean();
```

메시지 스트림에서 부울 값을 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 부울 값입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - 바이트 읽기

**인터페이스:**

```
Int16   ReadSignedByte();
    Byte    ReadByte();
```

메시지 스트림에서 부호 있는 8비트 정수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 바이트입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadBytes* - 바이트 읽기

**인터페이스:**

```
Int32   ReadBytes(Byte[] array);
```

메시지 스트림에서 바이트 배열을 읽습니다.

**매개변수:**

**array (input)**
바이트 배열을 포함하는 읽은 버퍼를 나타내며 바이트 단위의 버퍼 길이입니다.

배열의 바이트 수가 버퍼 길이보다 작거나 같을 경우 전체 배열이 버퍼로 읽힙니다. 배열의 바이트 수가 버퍼 길이보다 클 경우 버퍼가 배열의 일부로 채워지고 내부 커서가 다음으로 읽을 바이트의 위치를 표시합니다. readBytes()에 대한 이후 호출은 배열에서 현재 커서 위치에서 시작하는 바이트를 읽습니다.

입력 시 널 포인터를 지정하면 해당 호출이 바이트 배열을 읽지 않고 건너 뜁니다.

**리턴값:**
버퍼로 읽어 들인 바이트 수입니다. 버퍼의 일부만 채워진 경우 값이 버퍼 길이보다 작으며, 이는 배열에 더 이상 읽을 바이트가 남아 있지 않음을 나타냅니다. 호출하기 전에 배열에서 읽을 나머지 바이트가 없을 경우 값은 XMSC_END_OF_BYTEARRAY입니다.

입력 시 널 포인터를 지정하면 이 메소드가 값을 리턴하지 않습니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadChar* - 문자 읽기

**인터페이스:**

```
Char    ReadChar();
```

메시지 스트림에서 2바이트 문자를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 문자입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadDouble* - 배 정밀도 부동 소수점 수 읽기

**인터페이스:**

```
Double  ReadDouble();
```

메시지 스트림에서 8바이트 배 정밀도 부동 소수점 수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 배 정밀도 부동 소수점 수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadFloat* - 부동 소수점 수 읽기

**인터페이스:**

```
Single  ReadFloat();
```

메시지 스트림에서 4바이트 부동 소수점 수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 부동 소수점 수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 정수 읽기

**인터페이스:**

```
Int32   ReadInt();
```

메시지 스트림에서 부호 있는 32비트 정수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadLong - Long* 정수 읽기

**인터페이스:**

```
Int64   ReadLong();
```

메시지 스트림에서 부호 있는 64비트 정수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 Long 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadObject* - 오브젝트 읽기

**인터페이스:**

```
Object  ReadObject();
```

메시지 스트림에서 값을 읽은 후 해당 값의 데이터 유형을 리턴합니다.

**매개변수:**
없음

**리턴값:**
다음 오브젝트 유형 중 하나인 값입니다.

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

**예외:**
XMSException

*ReadShort - Short* 정수 읽기

**인터페이스:**

```
Int16   ReadShort();
```

메시지 스트림에서 부호 있는 16비트 정수를 읽습니다.

**매개변수:**
없음

**리턴값:**
읽은 Short 정수입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadString* - 문자열 읽기

**인터페이스:**

```
String  ReadString();
```

메시지 스트림에서 문자열을 읽습니다. 필요한 경우 XMS에서 문자열의 문자를 로컬 코드 페이지로 변환합니다.

**매개변수:**
없음

**리턴값:**
읽은 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 매개변수는 변환 후 문자열입니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*Reset* - 재설정

**인터페이스:**

```
void  Reset();
```

메시지 본문을 읽기 전용 모드로 전환하고 커서를 메시지 스트림의 시작 지점으로 옮깁니다.

**매개변수:**
없음

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotReadableException
- MessageEOFException

*WriteBoolean* - 부울 값 쓰기

**인터페이스:**

```
void  WriteBoolean(Boolean value);
```

부울 값을 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 부울 값입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteByte* - 바이트 쓰기

**인터페이스:**

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

바이트를 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 바이트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteBytes* - 바이트 쓰기

**인터페이스:**

```
void  WriteBytes(Byte[] value);
```

바이트 배열을 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 바이트 배열입니다.

**length (input)**
배열의 바이트 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteChar* - 문자 쓰기

**인터페이스:**

```
void  WriteChar(Char value);
```

문자를 2바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 문자입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteDouble* - 배 정밀도 부동 소수점 수 쓰기

**인터페이스:**

```
void  WriteDouble(Double value);
```

배 정밀도 부동 소수점 수를 long 정수로 변환하고 long 정수를 8바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 배 정밀도 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteFloat* - 부동 소수점 수 쓰기

**인터페이스:**

```
void  WriteFloat(Single value);
```

부동 소수점 수를 정수로 변환하고 정수를 4바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 부동 소수점 수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteInt* - 정수 쓰기

**인터페이스:**

```
void  WriteInt(Int32 value);
```

정수를 4바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteLong* - *Long* 정수 쓰기

**인터페이스:**

```
void  WriteLong(Int64 value);
```

long 정수를 8바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 Long 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteObject* - 오브젝트 쓰기

**인터페이스:**

```
void  WriteObject(Object value);
```

지정된 데이터 유형의 값을 메시지 스트림에 기록합니다.

**매개변수:**

**objectType (input)**
다음 오브젝트 유형 중 하나인 값입니다.

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

**value (input)**
기록할 값을 포함하는 바이트 배열입니다.

**length (input)**
배열의 바이트 수입니다.

**리턴값:**
Void

**예외:**

- XMSException

*WriteShort* - *Short* 정수 쓰기

**인터페이스:**

```
void  WriteShort(Int16 value);
```

short 정수를 2바이트(상위 바이트 우선)로 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 Short 정수입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

*WriteString* - 문자열 쓰기

**인터페이스:**

```
void  WriteString(String value);
```

문자열을 메시지 스트림에 기록합니다.

**매개변수:**

**value (input)**
기록할 문자열을 캡슐화하는 String 오브젝트입니다.

**리턴값:**
Void

**예외:**

- XMSException
- MessageNotWritableException

## 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

clearBody, clearProperties, PropertyExists

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

# ITextMessage

텍스트 메시지는 본문이 문자열로 구성된 메시지입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.ITextMessage
```

**관련 참조**
텍스트 메시지
텍스트 메시지의 본문은 문자열을 포함합니다.

## *.NETproperties*

*Text* - 텍스트 가져오기 및 설정

**인터페이스:**

```
String Text
{
  get;
```

```
  set;
}
```

텍스트 메시지의 본문을 구성하는 문자열을 가져오고 설정합니다.

필요한 경우 XMS에서 문자열의 문자를 로컬 코드 페이지로 변환합니다.

**예외:**

- XMSException
- MessageNotReadableException
- MessageNotWritableException
- MessageEOFException

### 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType, Properties

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

clearBody, clearProperties, PropertyExists

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## TransactionInProgressException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.TransactionInProgressException
```

XMS 트랜잭션이 진행 중이므로 애플리케이션이 유효하지 않은 조작을 요청하는 경우 이 예외가 발생합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## TransactionRolledBackException

**상속 계층 구조:**

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.TransactionRolledBackException
```

애플리케이션이 Session.commit()를 호출하여 현재 트랜잭션을 커미트하지만, 트랜잭션이 롤백된 경우 XMS는 이 예외를 처리합니다.

### 상속된 특성 및 메소드

다음 메소드는 XMSException 인터페이스에서 상속됩니다.

GetErrorCode, GetLinkedException

## XMSException

XMS가 .NET 메소드에 대한 호출을 처리하는 중 오류를 발견한 경우 XMS는 오류를 처리합니다. 예외는 오류 정보를 캡슐화하는 오브젝트입니다.

**상속 계층 구조:**

```
System.Exception
   |
   +----IBM.XMS.XMSException
```

다양한 유형의 XMS 예외가 있으며 XMSException 오브젝트는 이 중 한 가지 예외 유형입니다. 그러나 XMSException 클래스는 다른 XMS 예외 클래스의 수퍼클래스입니다. XMS는 다른 유형의 예외가 적합하지 않은 경우에 XMSException 오브젝트를 처리합니다.

### *.NETproperties*

*ErrorCode* - 오류 코드 가져오기

**인터페이스:**

```
public String ErrorCode
{
  get {return errorCode_;}
}
```

오류 코드를 가져옵니다.

**예외:**

- XMSException

*LinkedException* - 링크된 예외 가져오기

**인터페이스:**

```
public Exception LinkedException
{
  get { return linkedException_;}
  set { linkedException_ = value;}
}
```

예외 체인에서 다음 예외를 가져옵니다.

체인에 더 이상 예외가 없으면 메소드는 널을 리턴합니다.

**예외:**

- XMSException

## XMSFactoryFactory

애플리케이션이 관리 오브젝트를 사용하지 않는 경우 이 클래스를 사용하여 연결 팩토리, 큐 및 토픽을 작성합니다.

**상속 계층 구조:**
없음

### *.NETproperties*

*Metadata* - 메타데이터 검색

**인터페이스:**

```
IConnectionMetaData MetaData
```

XMSFactoryFactory 오브젝트의 연결 유형에 적합한 메타데이터를 가져옵니다.

**예외:**
없음

## 방법

*CreateConnectionFactory* - 연결 팩토리 작성

**인터페이스:**

```
IConnectionFactory CreateConnectionFactory();
```

선언된 유형의 ConnectionFactory 오브젝트를 작성합니다.

**매개변수:**
없음

**리턴값:**
ConnectionFactory 오브젝트입니다.

**예외:**

- XMSException

*CreateQueue* - 큐 작성

**인터페이스:**

```
IDestination CreateQueue(String name);
```

메시지 서버의 큐를 나타내는 Destination 오브젝트를 작성합니다.

이 메소드는 메시지 서버에 큐를 작성하지 않습니다. 따라서 애플리케이션이 이 메소드를 호출하려면 먼저 큐를 작성해야 합니다.

**매개변수:**

**name (input)**
큐 이름을 캡슐화하거나 큐를 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**
큐를 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*CreateTopic* - 토픽 작성

**인터페이스:**

```
IDestination CreateTopic(String name);
```

토픽을 나타내는 Destination 오브젝트를 작성합니다.

**매개변수:**

**name (input)**
토픽 이름을 캡슐화하거나 토픽을 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**
토픽을 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSException

*GetInstance* - *XMSFactoryFactory*의 인스턴스 가져오기

**인터페이스:**

```
static XMSFactoryFactory GetInstance(int connectionType);
```

XMSFactoryFactory의 인스턴스를 작성합니다. XMS 애플리케이션은 XMSFactoryFactory 오브젝트를 사용하여 필요한 프로토콜 유형에 적합한 ConnectionFactory 오브젝트에 대한 참조를 가져옵니다. 그러면, 이 ConnectionFactory 오브젝트는 해당 프로토콜 유형에 대한 연결만 생성할 수 있습니다.

**매개변수:**

**connectionType (input)**
ConnectionFactory 오브젝트가 연결을 생성하는 연결 유형입니다.

- XMSC.CT_WPM
- XMSC.CT_RTT
- XMSC.CT_WMQ

**리턴값:**
선언된 연결 유형에만 사용되는 XMSFactoryFactory 오브젝트입니다.

**예외:**

- NotSupportedException

# XMS 오브젝트 특성

이 장에서는 XMS에 의해 정의된 오브젝트 특성에 대해 설명합니다.

장에는 다음 절가 포함됩니다.

- 182 페이지의 『Session 특성』

각 절에는 지정된 유형의 오브젝트 특성이 나열되어 있으며 각 특성에 대한 간단한 설명이 제공됩니다.

이 장은 182 페이지의 『특성 정의』 절를 포함하며 각 특성의 정의를 제공합니다.

애플리케이션이 이 장에 설명된 오브젝트의 고유한 특성을 정의하는 경우, 오류가 발생하지 않지만 예측 불가능한 결과를 야기할 수 있습니다.

**참고:** 이 섹션의 특성 이름과 값은 C 및 C++에 사용되는 양식인 XMSC.*NAME* 양식으로 표시됩니다. 그러나 .NET에서 특성 이름의 양식은 사용하는 방법에 따라 XMSC.*NAME* 또는 XMSC_*NAME*일 수 있습니다.

- 특성을 지정하는 경우 특성 이름은 다음 예에 표시된 대로 XMSC.*NAME* 양식이어야 합니다.

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- 문자열을 지정하는 경우 특성 이름은 다음 예에 표시된대로 XMSC_*NAME* 양식이어야 합니다.

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

  .NET에서는 특성 이름 및 값이 XMSC 클래스의 상수로 제공됩니다. 이 상수는 문자열을 식별하며 XMS .NET 애플리케이션에서 사용합니다. 사전정의된 상수를 사용하는 경우 특성 이름과 값은 XMSC.*NAME* 양식으로 사용됩니다. 예를 들어 XMSC_USERID가 아닌 XMSC.USERID를 사용합니다.

또한 데이터 유형은 C/C++에 사용되는 형식입니다.41 페이지의 『.NET의 데이터 유형』에서 .NET 에 해당하는 값을 찾을 수 있습니다.

**관련 개념**

자체 애플리케이션 빌드

샘플 애플리케이션을 빌드하는 것처럼 자체 애플리케이션을 빌드할 수 있습니다.

**관련 참조**

.NET 인터페이스

이 절에서는 .NET 클래스 인터페이스와 해당 특성 및 메소드에 대해 설명합니다.

## Connection 특성

Connection 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 *26. Connection* 특성

| 특성 이름 | 설명 |
|---|---|
| 212 페이지의 『XMSC_WMQ_RESOLVED_QUEUE_MANAGER』 | 이 특성을 사용하여 연결된 큐 관리자의 이름을 얻을 수 있습니다. |
| 213 페이지의 『XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID』 | 이 특성은 연결 후에 큐 관리자의 ID로 채워집니다. |
| XMSC_WPM_CONNECTION_PROTOCOL | 메시징 엔진에 연결하는 데 사용되는 통신 프로토콜입니다. 이 특성은 읽기 전용입니다. |
| XMSC_WPM_HOST_NAME | 애플리케이션이 연결되는 메시징 엔진이 포함된 시스템의 호스트 이름 또는 IP 주소입니다. 이 특성은 읽기 전용입니다. |
| XMSC_WPM_ME_NAME | 애플리케이션이 연결되는 메시징 엔진의 이름입니다. 이 특성은 읽기 전용입니다. |
| XMSC_WPM_PORT | 애플리케이션이 연결되는 메시징 엔진이 대기하는 포트의 번호입니다. 이 특성은 읽기 전용입니다. |

연결 오브젝트에도 연결하기 위해 사용된 연결 팩토리의 특성에서 파생된 읽기 전용 특성이 있습니다. 이러한 특성은 연결될 시점에 설정된 연결 팩토리 특성에서 파생된 것 뿐만 아니라 설정되지 않은 특성의 기본값에서 파생

된 것도 있습니다. 특성에는 애플리케이션이 연결된 메시징 서버의 유형과 관련된 특성만 포함됩니다. 특성의 이름은 연결 팩토리 특성의 이름과 동일합니다.

## ConnectionFactory의 특성

ConnectionFactory 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

| 표 *27. ConnectionFactory*의 특성 | |
|---|---|
| **특성 이름** | **설명** |
| 191 페이지의 『XMSC_ASYNC_EXCEPTIONS』 | 이 특성은 연결이 끊기거나 XMS API 호출에 비동기적으로 예외가 발생하는 경우에만 XMS에서 ExceptionListener에 알리는지 여부를 판별합니다. 이 특성은 등록된 ExceptionListener가 있고 이 ConnectionFactory에서 작성된 모든 연결에 적용됩니다. |
| XMSC_CLIENT_ID | 연결에 사용하는 클라이언트 ID입니다. |
| XMSC_CONNECTION_TYPE | 애플리케이션이 연결되는 메시징 서버의 유형입니다. |
| XMSC_PASSWORD | 애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 비밀번호입니다. |
| 196 페이지의 『XMSC_RTT_BROKER_PING_INTERVAL』 | 이 시간 간격(밀리초) 후 XMS .NET에서 활동을 감지하기 위해 실시간 메시징 서버에 대한 연결을 검사합니다. |
| XMSC_RTT_CONNECTION_PROTOCOL | 브로커에 대한 실시간 연결에 사용되는 통신 프로토콜입니다. |
| XMSC_RTT_HOST_NAME | 브로커가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다. |
| XMSC_RTT_LOCAL_ADDRESS | 브로커에 대한 실시간 연결에 사용되는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다. |
| XMSC_RTT_MULTICAST | 연결 팩토리 또는 목적지의 멀티캐스트 설정입니다. |
| XMSC_RTT_PORT | 브로커가 수신 요청을 청취하는 포트 번호입니다. |
| XMSC_USERID | 애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 사용자 ID입니다. |
| XMSC_WMQ_BROKER_CONTROLQ | 브로커가 사용하는 제어 큐의 이름입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_BROKER_PUBQ | 애플리케이션이 발행하는 메시지를 송신하는 브로커에서 모니터하는 큐의 이름입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |

| 표 27. *ConnectionFactory*의 특성 (계속) | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_WMQ_BROKER_QMGR | 브로커가 연결되는 큐 관리자의 이름입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_BROKER_SUBQ | 지속 불가능한 메시지 이용자의 구독자 큐 이름입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_BROKER_VERSION | 애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| 200 페이지의 『XMSC_WMQ_CCDTURL』 | 클라이언트 채널 정의 테이블을 포함하는 파일의 이름과 위치를 식별하고 이 파일의 액세스 방법을 지정하는 URL(Uniform Resource Locator)입니다. |
| XMSC_WMQ_CHANNEL | 연결에 사용되는 채널 이름입니다. |
| 201 페이지의 『XMSC_WMQ_CLIENT_RECONNECT_OPTIONS』 | 이 특성은 이 팩토리에서 작성된 새 연결에 대한 클라이언트 다시 연결 옵션을 지정합니다. |
| 201 페이지의 『XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT』 | 이 특성은 클라이언트 연결이 다시 연결하려고 시도하는 시간(초)을 지정합니다. |
| XMSC_WMQ_CONNECTION_MODE | 애플리케이션이 큐 관리자에 연결할 때 사용되는 모드입니다. |
| 202 페이지의 『XMSC_WMQ_CONNECTION_NAME_LIST』 | 이 특성은 연결이 끊어진 후 클라이언트가 다시 연결하려고 시도하는 호스트를 지정합니다. |
| XMSC_WMQ_FAIL_IF_QUIESCE | 애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다. |
| XMSC_WMQ_HOST_NAME | 큐 관리자가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다. |
| XMSC_WMQ_LOCAL_ADDRESS | 큐 관리자에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다. |

| 표 27. *ConnectionFactory*의 특성 (계속) | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_WMQ_MESSAGE_SELECTION | 메시지 선택이 XMS 클라이언트 또는 브로커에서 수행되는지를 판별합니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_MSG_BATCH_SIZE | 비동기 메시지 전달을 사용할 때 하나의 일괄처리로 큐에서 검색되는 최대 메시지 수입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_POLLING_INTERVAL | 세션 내에 있는 각 메시지 리스너의 큐에 적합한 메시지가 없는 경우, 이 값은 각 메시지 리스너가 해당 큐에서 메시지 가져오기를 다시 시도하기 전에 경과하는 최대 간격(밀리초)입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| 210 페이지의 『XMSC_WMQ_PROVIDER_VERSION』 | 애플리케이션이 연결하려고 하는 큐 관리자의 버전, 릴리스, 수정 레벨, 수정팩입니다. |
| XMSC_WMQ_PORT | 큐 관리자 리스너가 수신 요청을 청취하는 포트 번호입니다. |
| XMSC_WMQ_PUB_ACK_INTERVAL | XMS 클라이언트가 브로커에서 수신확인을 요청하기 전에 공개자가 공개한 메시지 수입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| 206 페이지의 『XMSC_WMQ_PUT_ASYNC_ALLOWED』 | 이 특성은 메시지 생성자가 비동기 Put을 사용하여 메시지를 이 목적지로 송신할 수 있는지 여부를 판별합니다. |
| XMSC_WMQ_QMGR_CCSID | XMS 클라이언트와 WebSphere MQ 클라이언트 사이에서 교환되는 MQI(Message Queue Interface)에서 정의된 문자 데이터의 필드가 있는 코딩된 문자-세트의 ID(CCSID) 또는 코드 페이지입니다. |
| XMSC_WMQ_QUEUE_MANAGER | 연결된 큐 관리자의 이름. |
| XMSC_WMQ_RECEIVE_EXIT | 실행할 채널 수신 종료를 식별합니다. |
| XMSC_WMQ_RECEIVE_EXIT_INIT | 호출 시 채널 수신 종료에 전달되는 사용자 데이터입니다. |

| 표 27. *ConnectionFactory*의 특성 (계속) | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_WMQ_SECURITY_EXIT | 채널 보안 엑시트를 식별합니다. |
| XMSC_WMQ_SECURITY_EXIT_INIT | 호출 시 채널 보안 엑시트로 전달되는 사용자 데이터입니다. |
| 214 페이지의 『XMSC_WMQ_SEND_CHECK_COUNT』 | 변환되지 않은 하나의 XMS 세션에서 비동기 put 오류를 검사하는 사이에 허용할 수 있는 송신 호출 수입니다. |
| XMSC_WMQ_SEND_EXIT | 채널 전송 종료를 식별합니다. |
| XMSC_WMQ_SEND_EXIT_INIT | 호출 시 채널 송신 엑시트로 전달되는 사용자 데이터입니다. |
| 214 페이지의 『XMSC_WMQ_SHARE_CONV_ALLOWED』 | 클라이언트 연결이 채널 정의가 일치하는 경우 동일한 프로세스에서 동일한 큐 관리자로의 다른 최상위 레벨 XMS 연결과 소켓을 공유할 수 있는지 여부입니다. 이 특성은 애플리케이션 개발 또는 유지보수 또는 경영 목적을 위해 필요한 경우 별도의 소켓에서 연결의 완전한 격리를 허용하기 위해 제공됩니다. |
| XMSC_WMQ_SSL_CERT_STORES | 큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 인증서 폐기 목록(CRL)이 있는 서버의 위치입니다. |
| XMSC_WMQ_SSL_CIPHER_SPEC | 큐 관리자에 대한 보안 연결에 사용할 CipherSpec의 이름. |
| XMSC_WMQ_SSL_CIPHER_SUITE | 큐 관리자에 대한 SSL 또는 TLS 연결에 사용될 CipherSuite 의 이름입니다. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다. |
| XMSC_WMQ_SSL_CRYPTO_HW | 클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항입니다. |
| XMSC_WMQ_SSL_FIPS_REQUIRED | 이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 또는 사용할 수 없는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 FIPS 알고리즘만 클라이언트-서버 연결에 사용됩니다. |
| XMSC_WMQ_SSL_KEY_REPOSITORY | 키 및 인증서가 저장되는 키 데이터베이스 파일의 위치입니다. |
| XMSC_WMQ_SSL_KEY_RESETCOUNT | KeyResetCount는 비밀 키가 재협상되기 전에 SSL 통신에서 송신되고 수신된 암호화되지 않은 총 바이트 수를 나타냅니다. |
| XMSC_WMQ_SSL_PEER_NAME | 큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 피어 이름입니다. |
| | |
| XMSC_WMQ_SYNCPOINT_ALL_GETS | 동기점 제어에 있는 큐에서 모든 메시지를 검색해야 하는지 여부를 결정합니다. |
| 221 페이지의 『XMSC_WMQ_TARGET_CLIENT』 | |
| XMSC_WMQ_TEMP_Q_PREFIX | 애플리케이션이 XMS 임시 큐를 작성할 때 작성되는 WebSphere MQ 동적 큐의 이름을 구성하는 데 사용되는 접두부입니다. |
| XMSC_WMQ_TEMP_TOPIC_PREFIX | 임시 주제를 작성할 때 XMS 는 "TEMP/TEMPTOPICPREFIX/unique_id" 양식의 주제 문자열을 생성하거나 이 특성에 기본값이 포함된 경우 "TEMP/unique_id" 문자열이 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다. |
| XMSC_WMQ_TEMPORARY_MODEL | 애플리케이션이 XMS 임시 큐를 작성할 때 동적 큐가 작성되는 WebSphere MQ 모델 큐의 이름입니다. |

| 표 27. *ConnectionFactory*의 특성 (계속) | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_WPM_BUS_NAME | 연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다. |
| XMSC_WPM_CONNECTION_PROXIMITY | 연결에 대한 연결 근접성 설정입니다. |
| XMSC_WPM_DUR_SUB_HOME | 목적지나 연결에 대한 모든 지속 가능한 구독을 관리하는 메시징 엔진의 이름입니다. |
| XMSC_WPM_LOCAL_ADDRESS | 서비스 통합 버스에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다. |
| XMSC_WPM_NON_PERSISTENT_MAP | 연결을 사용하여 송신되는 비지속적 메시지의 신뢰도 레벨입니다. |
| XMSC_WPM_PERSISTENT_MAP | 연결을 사용하여 송신되는 지속 메시지의 신뢰도 레벨입니다. |
| XMSC_WPM_PROVIDER_ENDPOINTS | 부트스트랩 서버에 사용되는 하나 이상의 엔드포인트 주소의 순서입니다. |
| XMSC_WPM_TARGET_GROUP | 메시징 엔진의 대상 그룹 이름입니다. |
| XMSC_WPM_TARGET_SIGNIFICANCE | 메시징 엔진의 대상 그룹에 대한 중요성입니다. |
| XMSC_WPM_TARGET_TRANSPORT_CHAIN | 애플리케이션이 메시징 엔진에 연결하는 데 사용해야 하는 인바운드 전송 체인의 이름입니다. |
| XMSC_WPM_TARGET_TYPE | 메시징 엔진의 대상 그룹 유형입니다. |
| XMSC_WPM_TEMP_Q_PREFIX | 애플리케이션이 XMS 임시 큐를 작성할 때 서비스 통합 버스에 작성되는 임시 큐의 이름을 구성하는 데 사용되는 접두부입니다. |
| XMSC_WPM_TEMP_TOPIC_PREFIX | 애플리케이션이 작성한 임시 토픽의 이름을 구성하는 데 사용되는 접두부입니다. |

**관련 개념**

ConnectionFactories 오브젝트 및 Connection 오브젝트
ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

WebSphere MQ 큐 관리자에 대한 보안 연결
XMS .NET 애플리케이션이 WebSphere MQ 큐 관리자에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

WebSphere 서비스 통합 버스 메시징 엔진에 대한 보안 연결
XMS 애플리케이션이 WebSphere 서비스 통합 버스 메시징 엔진에 보안 연결하려면, 관련 특성을 ConnectionFactory 오브젝트에 정의해야 합니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

**관련 태스크**

관리 대상 오브젝트 작성

XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**

관리 대상 ConnectionFactory 오브젝트의 필수 특성
애플리케이션이 연결 팩토리를 작성하는 경우 메시징 서버로의 연결을 작성하려면 여러 가지 특성이 정의되어야 합니다.

## ConnectionMetaData 특성

ConnectionMetaData 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 *28. ConnectionMetaData* 특성

| 특성 이름 | 설명 |
|---|---|
| XMSC_JMS_MAJOR_VERSION | XMS의 기반이 되는 JMS 스펙의 주 버전 번호입니다. 이 특성은 읽기 전용입니다. |
| XMSC_JMS_MINOR_VERSION | XMS의 기반이 되는 JMS 스펙의 부 버전 번호입니다. 이 특성은 읽기 전용입니다. |
| XMSC_JMS_VERSION | XMS의 기반이 되는 JMS 스펙의 부 버전 번호입니다. 이 특성은 읽기 전용입니다. |
| XMSC_MAJOR_VERSION | XMS 클라이언트의 버전 번호입니다. 이 특성은 읽기 전용입니다. |
| XMSC_MINOR_VERSION | XMS 클라이언트의 릴리스 번호입니다. 이 특성은 읽기 전용입니다. |
| XMSC_PROVIDER_NAME | XMS 클라이언트의 제공자입니다. 이 특성은 읽기 전용입니다. |
| XMSC_VERSION | XMS 클라이언트의 버전 ID입니다. 이 특성은 읽기 전용입니다. |

## Destination 특성

Destination 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 *29. Destination* 특성

| 특성 이름 | 설명 |
|---|---|
| XMSC_DELIVERY_MODE | 목적지에 전송되는 메시지의 전달 모드입니다. |
| XMSC_PRIORITY | 목적지에 전송되는 메시지의 우선순위입니다. |
| XMSC_RTT_MULTICAST | 연결 팩토리 또는 목적지의 멀티캐스트 설정입니다. |
| XMSC_TIME_TO_LIVE | 목적지에 송신되는 메시지의 TTL(Time to Live). |
| XMSC_WMQ_BROKER_VERSION | 애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다. |
| XMSC_WMQ_CCSID | XMS 클라이언트에서 메시지를 목적지로 전달할 때 메시지 본문에 있는 문자 데이터의 문자열에 사용되는 코드화된 문자 세트 ID(CCSID) 또는 코드 페이지입니다. |

표 *29. Destination* 특성 *(계속)*

| 특성 이름 | 설명 |
| --- | --- |
| XMSC_WMQ_DUR_SUBQ | 목적지에서 메시지를 수신하는 지속 가능한 구독자의 구독자 큐 이름입니다.<br>**참고:** 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다. |
| XMSC_WMQ_ENCODING | XMS 클라이언트가 메시지를 목적지에 전달할 때 메시지 본문의 숫자 데이터를 표시하는 방법입니다. |
| XMSC_WMQ_FAIL_IF_QUIESCE | 애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다. |
| 204 페이지의 『XMSC_WMQ_MESSAGE_BODY』 | 이 특성은 XMS 애플리케이션이 메시지 페이로드의 일부(즉, 메시지 본문의 일부)로서 IBM WebSphere MQ 메시지의 MQRFH2를 처리하는지 여부를 판별합니다. |
| 204 페이지의 『XMSC_WMQ_MQMD_MESSAGE_CONTEXT』 | XMS 애플리케이션이 설정할 메시지 컨텍스트의 레벨을 결정합니다. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다. |
| 205 페이지의 『XMSC_WMQ_MQMD_READ_ENABLED』 | 이 특성은 XMS 애플리케이션이 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다. |
| 206 페이지의 『XMSC_WMQ_MQMD_WRITE_ENABLED』 | 이 특성은 XMS 애플리케이션이 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다. |
| 206 페이지의 『XMSC_WMQ_READ_AHEAD_ALLOWED』 | 이 특성은 메시지를 수신하기 전에 메시지 이용자와 큐 브라우저가 이 목적지에서 내부 버퍼로 트랜잭트되지 않은 비지속적 메시지를 가져오기 위해 미리 읽기를 사용할 수 있는지 여부를 판별합니다. |
| 207 페이지의 『XMSC_WMQ_READ_AHEAD_CLOSE_POLICY』 | 비동기 메시지 리스너로 전달되는 메시지의 경우, 메시지 이용자를 닫을 때 내부 미리 읽기 버퍼에서 메시지에 일어나는 사항을 이 특성에서 판별합니다. |
| 211 페이지의 『XMSC_WMQ_RECEIVE_CCSID』 | 큐 관리자 메시지 변환용 대상 CCSID를 설정하는 목적지 특성입니다. XMSC_WMQ_RECEIVE_CONVERSION을 WMQ_RECEIVE_CONVERSION_QMGR로 설정하지 않은 경우, 값이 무시됩니다. |
| 212 페이지의 『XMSC_WMQ_RECEIVE_CONVERSION』 | 큐 관리자가 데이터 변환을 수행할지 판별하는 목적지 특성입니다. |
| XMSC_WMQ_TARGET_CLIENT | 목적지에 송신되는 메시지에 MQRFH2 헤더를 포함할지 여부를 지정합니다. |
| XMSC_WMQ_TEMP_TOPIC_PREFIX | 임시 주제를 작성할 때 XMS 는 "TEMP/TEMPTOPICPREFIX/unique_id" 양식의 주제 문자열을 생성하거나 이 특성에 기본값이 포함된 경우 "TEMP/unique_id" 문자열이 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다. |

| 표 *29. Destination* 특성 *(계속)* | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_WPM_BUS_NAME | 연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다. |
| XMSC_WPM_TOPIC_SPACE | 토픽을 포함하는 토픽 영역의 이름입니다. |

**관련 개념**

ConnectionFactories 오브젝트 및 Connection 오브젝트
ConnectionFactory 오브젝트는 애플리케이션에서 Connection 오브젝트를 작성하는 데 사용하는 템플리트를 제공합니다. 애플리케이션은 Connection 오브젝트를 사용하여 Session 오브젝트를 작성합니다.

WebSphere 서비스 통합 버스에 대한 연결
직접 TCP/IP 연결을 사용하거나 TCP/IP에서 HTTP를 사용하여 XMS 애플리케이션을 WebSphere 서비스 통합 버스에 연결할 수 있습니다.

목적지
XMS 애플리케이션은 Destination 오브젝트를 사용하여 전송 중인 메시지의 목적지와 수신 중인 메시지의 소스를 지정합니다.

목적지 와일드 카드
XMS는 목적지 와일드 카드를 지원하므로 일치를 위해 필요한 위치로 와일드 카드를 전달할 수 있습니다. XMS가 작업할 수 있는 각 서버 유형마다 와일드 카드 설계가 다릅니다.

토픽 URI(Uniform Resource Identifier)
토픽 URI(Uniform Resource Identifier)는 토픽의 이름을 지정합니다. 또한, 토픽 특성을 한 개 이상 지정할 수 있습니다.

큐 URI(Uniform Resource Identifier)
큐 URI는 큐의 이름을 지정합니다. 또한, 큐 특성을 한 개 이상 지정할 수 있습니다.

임시 목적지
XMS 애플리케이션은 임시 목적지를 작성하고 사용할 수 있습니다.

관리 대상 오브젝트의 특성 맵핑
애플리케이션에서 WebSphere MQ JMS 및 WebSphere Application Server 연결 팩토리 및 대상 오브젝트 정의를 사용하려면, 이 정의에서 검색된 특성을 XMS 연결 팩토리 및 목적지에서 설정할 수 있는 해당 XMS 특성에 맵핑해야 합니다.

**관련 태스크**

관리 대상 오브젝트 작성
XMS 애플리케이션이 메시징 서버에 연결하는 데 필요한 ConnectionFactory 및 Destination 오브젝트 정의는 적절한 관리 도구를 사용하여 작성해야 합니다.

**관련 참조**

관리 대상 Destination 오브젝트의 필수 특성
목적지를 작성하는 애플리케이션은 관리 대상 Destination 오브젝트에 대한 여러 특성을 설정해야 합니다.

## InitialContext 특성

InitialContext 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

| 표 *30. InitialContext* 특성 | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_IC_PROVIDER_URL | 이를 사용하여 JNDI 네이밍 디렉토리를 찾으므로 COS 네이밍 서비스가 웹 서비스와 동일한 서버에 있을 필요가 없습니다. |
| XMSC_IC_SECURITY_AUTHENTICATION | Java 컨텍스트 인터페이스 SECURITY_AUTHENTICATION을 기반으로 합니다. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다. |

| 표 *30. InitialContext* 특성 *(계속)* | |
|---|---|
| **특성 이름** | **설명** |
| XMSC_IC_SECURITY_CREDENTIALS | Java 컨텍스트 인터페이스 SECURITY_CREDENTIALS를 기반으로 합니다. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다. |
| XMSC_IC_SECURITY_PRINCIPAL | Java 컨텍스트 인터페이스 SECURITY_PRINCIPAL을 기반으로 합니다. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다. |
| XMSC_IC_SECURITY_PROTOCOL | Java 컨텍스트 인터페이스 SECURITY_PROTOCOL을 기반으로 합니다. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다. |
| XMSC_IC_URL | LDAP 및 FileSystem 컨텍스트의 경우 관리 오브젝트가 들어 있는 저장소의 주소입니다. COS 네이밍 컨텍스트의 경우, 디렉토리의 오브젝트를 찾는 웹 서비스의 주소입니다. |

**관련 개념**

InitialContext 특성

InitialContext 생성자의 매개변수에는 URI(Uniform Resource Indicator) 형식으로 제공되는 관리 대상 오브젝트 저장소 위치가 포함됩니다. 애플리케이션에서 저장소에 연결하려면 URI에 포함된 정보 외에 추가 정보를 제공해야 합니다.

XMS 초기 컨텍스트의 URI 형식

관리 대상 오브젝트의 저장소 위치는 URI(Uniform Resource Indicator)로 제공됩니다. URI의 형식은 컨텍스트 유형에 따라 다릅니다.

관리 대상 오브젝트의 검색

XMS는 InitialContext 오브젝트가 작성될 때 제공된 주소 또는 InitialContext 특성의 주소를 사용하여 저장소에서 관리 대상 오브젝트를 검색합니다.

**관련 태스크**

InitialContext 오브젝트

애플리케이션은 필수 관리 대상 오브젝트를 검색할 수 있도록 관리 대상 오브젝트 저장소에 대한 연결을 작성하는 데 사용되는 초기 컨텍스트를 작성해야 합니다.

## Message 특성

Message 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

| 표 *31. Message* 특성 | |
|---|---|
| **특성 이름** | **설명** |
| JMS_IBM_CHARACTER_SET | XMS 클라이언트에서 메시지를 의도한 목적지로 전달할 때 메시지 본문에 있는 문자 데이터의 문자열에 사용되는 코드화된 문자 세트 ID(CCSID) 또는 코드 페이지입니다. XMS에서는 이 특성이 숫자 값이며 CCSID에 맵핑됩니다. 단, 이 특성은 JMS 특성을 기반으로 하므로 문자열 유형 값을 가지며 이 숫자 CCSID를 나타내는 Java 문자 세트에 맵핑됩니다. |
| JMS_IBM_ENCODING | XMS 클라이언트가 메시지를 의도한 목적지에 전달할 때 메시지 본문의 숫자 데이터를 표시하는 방법입니다. |
| JMS_IBM_EXCEPTIONMESSAGE | 메시지가 예외 목적지로 송신된 이유를 설명하는 텍스트입니다. 이 특성은 읽기 전용입니다. |
| JMS_IBM_EXCEPTIONPROBLEMDESTINATION | 메시지가 예외 목적지에 송신되기 전에 있었던 목적지 이름입니다. |

표 *31. Message* 특성 *(계속)*

| 특성 이름 | 설명 |
|---|---|
| JMS_IBM_EXCEPTIONREASON | 메시지가 예외 목적지에 송신된 이유를 나타내는 이유 코드입니다. |
| JMS_IBM_EXCEPTIONTIMESTAMP | 메시지가 예외 목적지로 전송된 시간입니다. |
| JMS_IBM_FEEDBACK | 보고서 메시지의 속성을 나타내는 코드입니다. |
| JMS_IBM_FORMAT | 메시지에서 애플리케이션 데이터의 속성입니다. |
| JMS_IBM_LAST_MSG_IN_GROUP | 메시지가 메시지 그룹에서 마지막 메시지인지 나타냅니다. |
| JMS_IBM_MSGTYPE | 메시지의 유형입니다. |
| JMS_IBM_PUTAPPLTYPE | 메시지를 송신한 애플리케이션의 유형입니다. |
| JMS_IBM_PUTDATE | 메시지가 송신된 날짜입니다. |
| JMS_IBM_PUTTIME | 메시지가 송신된 시간입니다. |
| JMS_IBM_REPORT_COA | 보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '도착 확인' 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_COD | 보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '전달 확인' 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_DISCARD_MSG | 메시지를 의도한 목적지에 전달할 수 없을 경우 메시지를 버리는 요청입니다. |
| JMS_IBM_REPORT_EXCEPTION | 보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 예외 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_EXPIRATION | 보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 만기 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_NAN | 부정적인 조치 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_PAN | 긍정적인 조치 알림 보고서 메시지를 요청합니다. |
| JMS_IBM_REPORT_PASS_CORREL_ID | 보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다. |
| JMS_IBM_REPORT_PASS_MSG_ID | 보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다. |
| JMS_IBM_RETAIN | 이 특성을 설정하면 메시지를 보유된 발행물로 처리하도록 큐 관리자에 표시합니다. |
| JMS_IBM_SYSTEM_MESSAGEID | 서비스 통합 버스 내에서 메시지를 고유하는 식별하는 ID입니다. 이 특성은 읽기 전용입니다. |
| JMSX_APPID | 메시지를 송신한 애플리케이션의 이름입니다. |
| JMSX_DELIVERY_COUNT | 메시지 전달 시도 횟수입니다. |
| JMSX_GROUPID | 메시지가 속한 메시지 그룹의 ID입니다. |
| JMSX_GROUPSEQ | 메시지 그룹에 있는 메시지의 순서 번호입니다. |
| JMSX_USERID | 메시지를 송신한 애플리케이션과 연관된 사용자 ID입니다. |

### JMS_IBM_MQMD* 특성

IBM Message Service Client for .NET을 사용하면 클라이언트 애플리케이션이 API를 사용하여 MQMD 필드를 읽고 쓸 수 있습니다. 또한 MQ 메시지 데이터에 액세스할 수도 있습니다. 기본적으로 MQMD에 대한 액세스는 사

용 불가능한 상태이므로 Destination 특성 XMSC_WMQ_MQMD_WRITE_ENABLED 및 XMSC_WMQ_MQMD_READ_ENABLED를 사용하여 애플리케이션이 명시적으로 사용하도록 설정해야 합니다. 이 두 특성은 서로 독립적입니다.

StrucId 및 Version을 제외한 모든 MQMD 필드는 추가 Message 오브젝트 특성으로 표시되며 JMS_IBM_MQMD가 접두부로 사용됩니다.

JMS_IBM_MQMD* 특성은 이전 표에 설명된 JMS_IBM*과 같은 다른 특성보다 더 우선합니다.

## 메시지 송신

StrucId 및 Version을 제외한 모든 MQMD 필드가 표시됩니다. 이러한 특성은 MQMD 필드만 참조합니다. 여기서 특성은 MQMD 및 MQRFH2 헤더 모두에서 발생하며, MQRFH2의 버전은 설정되거나 추출되지 않습니다. JMS_IBM_MQMD_BackoutCount를 제외한 이러한 특성을 설정할 수 있습니다. JMS_IBM_MQMD_BackoutCount에 대해 설정된 값은 무시됩니다.

특성이 최대 길이를 보유하며 너무 긴 값을 제공하는 경우에는 값이 잘립니다.

특정 특성의 경우, 목적지 오브젝트에서 XMSC_WMQ_MQMD_MESSAGE_CONTEXT 특성을 설정해야 할 수도 있습니다. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다. XMSC_WMQ_MQMD_MESSAGE_CONTEXT를 적합한 값으로 설정하지 않으면 특성 값이 무시됩니다. XMSC_WMQ_MQMD_MESSAGE_CONTEXT를 적합한 값으로 설정하지만 큐 관리자에 대해 충분한 컨텍스트 권한이 없는 경우에는 예외가 발생합니다. XMSC_WMQ_MQMD_MESSAGE_CONTEXT의 특정 값을 요구하는 특성은 다음과 같습니다.

다음 특성은 XMSC_WMQ_MQMD_MESSAGE_CONTEXT가 XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT 또는 XMSC_WMQ_MDCTX_SET_ALL_CONTEXT로 설정되도록 요구합니다.

- JMS_IBM_MQMD_UserIdentifier
- JMS_IBM_MQMD_AccountingToken
- JMS_IBM_MQMD_ApplIdentityData

다음 특성은 XMSC_WMQ_MQMD_MESSAGE_CONTEXT가 XMSC_WMQ_MDCTX_SET_ALL_CONTEXT로 설정되도록 요구합니다.

- JMS_IBM_MQMD_PutApplType
- JMS_IBM_MQMD_PutApplName
- JMS_IBM_MQMD_PutDate
- JMS_IBM_MQMD_PutTime
- JMS_IBM_MQMD_ApplOriginData

## 메시지 수신

생성 애플리케이션이 설정한 실제 특성과는 무관하게 XMSC_WMQ_MQMD_READ_ENABLED 특성이 true로 설정된 경우 이 모든 특성은 수신된 메시지에서 사용 가능합니다. JMS 스펙에 따라 우선 모든 특성이 선택 취소되지 않는 한 애플리케이션은 수신된 메시지의 특성을 수정할 수 없습니다. 수신된 메시지는 특성의 수정 없이 전달될 수 없습니다.

**참고:** XMSC_WMQ_MQMD_READ_ENABLED 특성이 true로 설정된 목적지에서 애플리케이션이 메시지를 수신하고 XMSC_WMQ_MQMD_WRITE_ENABLED가 true로 설정된 목적지에 이를 전달하는 경우, 이 결과에 따라 수신된 메시지의 모든 MQMD 필드 값은 전달된 메시지에 복사됩니다. 특성 표

표 *32. MQMD* 필드를 표시하는 *Message* 오브젝트

| 특성 | 설명 | 유형 |
|---|---|---|
| JMS_IBM_MQMD_REPORT | 보고 메시지의 옵션 | System.Int32 |
| JMS_IBM_MQMD_MSGTYPE | 메시지 유형 | System.Int32 |
| JMS_IBM_MQMD_EXPIRY | 메시지 수명 | System.Int32 |

표 *32. MQMD* 필드를 표시하는 *Message* 오브젝트 *(계속)*

| 특성 | 설명 | 유형 |
|---|---|---|
| JMS_IBM_MQMD_FEEDBACK | 피드백 또는 이유 코드 | System.Int32 |
| JMS_IBM_MQMD_ENCODING | 메시지 데이터의 숫자 인코딩 | System.Int32 |
| JMS_IBM_MQMD_CODEDCHARSETID | 메시지 데이터의 문자 세트 ID | System.Int32 |
| JMS_IBM_MQMD_FORMAT | 메시지 데이터의 형식 이름 | System.String |
| JMS_IBM_MQMD_PRIORITY<br>**참고:** 0-9 범위에 없는 JMS_IBM_MQMD_PRIORITY로 값이 지정되는 경우, 이 값은 JMS 스펙을 위반합니다. | 메시지 우선순위 | System.Int32 |
| JMS_IBM_MQMD_PERSISTENCE | 메시지 지속성 | System.Int32 |
| JMS_IBM_MQMD_MSGID<br>**참고:** JMS 스펙은 메시지 ID가 JMS 제공자에 의해 설정되어야 하며 고유하거나 널이어야 함을 지정합니다. 값을 JMS_IBM_MQMD_MSGID로 지정하는 경우, 이 값은 JMSMessageID에 복사됩니다. 따라서 이는 JMS 제공자에 의해 설정되지 않으며 고유하지 않을 수 있습니다. 이 값은 JMS 스펙을 위반합니다. | 메시지 ID | 바이트 배열<br>**참고:** 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다. |
| JMS_IBM_MQMD_CORRELID<br>**참고:** 'ID:' 문자열로 시작되는 JMS_IBM_MQMD_CORRELID로 값을 지정하는 경우, 이 값은 JMS 스펙을 위반합니다. | 상관 ID | 바이트 배열<br>**참고:** 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다. |
| JMS_IBM_MQMD_BACKOUTCOUNT | 백아웃 카운터 | System.Int32 |
| JMS_IBM_MQMD_REPLYTOQ | 응답 큐의 이름 | System.String |
| JMS_IBM_MQMD_REPLYTOQMGR | 응답 큐 관리자의 이름 | System.String |
| JMS_IBM_MQMD_USERIDENTIFIER | 사용자 ID | System.String |
| JMS_IBM_MQMD_ACCOUNTINGTOKEN | 계정 토큰 | 바이트 배열<br>**참고:** 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다. |
| JMS_IBM_MQMD_APPLIDENTITYDATA | ID 관련 애플리케이션 데이터 | System.String |
| JMS_IBM_MQMD_PUTAPPLTYPE | 메시지를 넣는 애플리케이션의 유형 | System.Int32 |
| JMS_IBM_MQMD_PUTAPPLNAME | 메시지를 보관하는 응용프로그램의 이름 | System.String |
| JMS_IBM_MQMD_PUTDATE | 메시지를 넣은 날짜 | System.String |
| JMS_IBM_MQMD_PUTTIME | 메시지를 넣은 시간 | System.String |
| JMS_IBM_MQMD_APPLORIGINDATA | 원본 관련 애플리케이션 데이터 | System.String |

| 표 32. *MQMD* 필드를 표시하는 *Message* 오브젝트 *(계속)* | | |
|---|---|---|
| **특성** | **설명** | **유형** |
| JMS_IBM_MQMD_GROUPID | 그룹 ID | 바이트 배열<br>**참고:** 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다. |
| JMS_IBM_MQMD_MSGSEQNUMBER | 그룹 내 로컬 메시지의 순서 번호 | System.Int32 |
| JMS_IBM_MQMD_OFFSET | 논리 메시지의 시작에서 실제 메시지의 데이터의 오프셋 | System.Int32 |
| JMS_IBM_MQMD_MSGFLAGS | 메시지 플래그 | System.Int32 |
| JMS_IBM_MQMD_ORIGINALLENGTH | 원래 메시지의 길이 | System.Int32 |

추가 세부사항은 MQMD를 참조하십시오.

### 예:

이 예제의 결과로 MQMD.UserIdentifier가 "JoeBloggs"로 설정된 큐 또는 토픽에 메시지를 넣습니다.

```
// 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_USERIDENTIFIER를 설정하기 전에 XMSC_WMQ_MQMD_MESSAGE_CONTEXT를 설정해야 합니다. XMSC_WMQ_MQMD_MESSAGE_CONTEXT 사용에 대한 자세한 정보는 Message 오브젝트 특성을 참조하십시오.

이와 유사하게, 메시지를 수신하기 전에 XMSC_WMQ_MQMD_READ_ENABLED를 true로 설정한 후에 메시지의 Get 메소드(예: getStringProperty)를 사용하여 MQMD 필드의 컨텐츠를 추출할 수 있습니다. 수신된 특성은 읽기 전용입니다.

이 예에서는 value 필드가 큐나 토픽에서 가져온 메시지의 MQMD.ApplIdentityData 필드의 값을 보유하게 됩니다.

```
// 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 특성

MessageConsumer 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 *33. MessageConsumer* 특성

| 특성 이름 | 설명 |
|---|---|
| XMSC_IS_SUBSCRIPTION_MULTICAST | 메시지가 WebSphere MQ 멀티캐스트 전송을 사용하여 메시지 이용자로 전달되는지 여부를 나타냅니다. 이 특성은 읽기 전용입니다. |
| XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST | 메시지가 WebSphere MQ 멀티캐스트 전송을 사용하여 신뢰할 수 있는 서비스 품질(QoS)로 메시지 이용자에게 전달되는지 여부를 나타냅니다. 이 특성은 읽기 전용입니다. |

자세한 정보는 IMessageConsumer의 .NET 특성을 참조하십시오.

## MessageProducer 특성

MessageProducer 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

자세한 내용은 IMessageProducer의 .NET 특성을 참조하십시오.

## Session 특성

Session 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

자세한 내용은 ISession의 .NET 특성을 참조하십시오.

## 특성 정의

이 절는 각 오브젝트 특성의 정의를 제공합니다.

각 특성 정의에는 다음 정보가 포함되어 있습니다.

- 등록 정보의 데이터 유형
- 특성이 있는 오브젝트의 유형
- Destination의 특성의 경우 URI(Uniform Resource Identifier)에서 사용할 수 있는 이름
- 특성의 자세한 설명
- 특성의 올바른 값
- 등록 정보의 기본값

이름이 다음 접두부 중 하나로 시작되는 특성은 지정된 연결 유형에 대해서만 관련됩니다.

**XMSC_RTT**
실시간 브로커 연결에 대해서만 특성이 관련됩니다. 특성 이름이 `xmsc_rtt.h` 헤더 파일의 이름 지정된 상수로 정의됩니다.

**XMSC_WMQ**
애플리케이션이 WebSphere MQ 큐 관리자에 연결되는 경우에만 특성이 관련됩니다. 특성 이름이 `xmsc_wmq.h` 헤더 파일의 이름 지정된 상수로 정의됩니다.

**XMSC_WPM**
애플리케이션이 WebSphere 서비스 통합 버스에 연결되는 경우에만 특성이 관련됩니다. 특성 이름이 `xmsc_wpm.h` 헤더 파일의 이름 지정된 상수로 정의됩니다.

정의에서 별도의 언급이 없는 한 나머지 특성은 모든 유형의 연결과 관련됩니다. 특성 이름이 `xmsc.h` 헤더 파일의 이름 지정된 상수로 정의됩니다. 접두부 MSX로 시작하는 이름의 특성은 메시지의 JMS 정의 특성이며, 접두부

JMS_IBM으로 시작하는 이름의 특성은 메시지의 IBM 정의 특성입니다. 메시지 특성에 대한 자세한 정보는 65 페이지의 『XMS 메시지의 특성』의 내용을 참조하십시오.

정의에서 별도의 언급이 없는 한, 각 특성은 포인트-투-포인트 및 발행/구독 도메인 둘 다와 관련됩니다.

특성이 읽기 전용으로 지정되지 않은 한 애플리케이션이 특성의 값을 가져와서 설정할 수 있습니다.

### *JMS_IBM_CHARACTER_SET*

**데이터 유형:**
System.Int32

**특성:**
메시지

XMS 클라이언트에서 메시지를 의도한 목적지로 전달할 때 메시지 본문에 있는 문자 데이터의 문자열에 사용되는 코드화된 문자 세트 ID(CCSID) 또는 코드 페이지입니다. XMS에서는 이 특성이 숫자 값이며 CCSID에 맵핑됩니다. 단, 이 특성은 JMS 특성을 기반으로 하므로 문자열 유형 값을 가지며 이 숫자 CCSID를 나타내는 Java 문자 세트에 맵핑됩니다. 이 특성은 목적지에 대해 지정된 CCSID를 XMSC_WMQ_CCSID 특성으로 대체합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_ENCODING*

**데이터 유형:**
System.Int32

**특성:**
메시지

XMS 클라이언트가 메시지를 의도한 목적지에 전달할 때 메시지 본문의 숫자 데이터를 표시하는 방법입니다. 이 특성은 목적지에 대해 지정된 인코딩을 XMSC_WMQ_ENCODING 특성으로 대체합니다. 이 특성은 2진 정수, 팩형 10진수 정수 및 부동 소수점 숫자의 표시를 지정합니다.

특성의 유효값은 메시지 디스크립터의 **Encoding** 필드에서 지정할 수 있는 값과 동일합니다.

애플리케이션은 다음과 같은 이름 지정된 상수를 사용하여 특성을 설정할 수도 있습니다.

| 이름 지정된 상수 | 의미 |
|---|---|
| MQENC_INTEGER_NORMAL | 표준 정수 인코딩 |
| MQENC_INTEGER_REVERSED | 역방향 정수 인코딩 |
| MQENC_DECIMAL_NORMAL | 표준 팩형 10진수 인코딩 |
| MQENC_DECIMAL_REVERSED | 역방향 팩형 10진수 인코딩 |
| MQENC_FLOAT_IEEE_NORMAL | 표준 IEEE 부동 소수점 인코딩 |
| MQENC_FLOAT_IEEE_REVERSED | 역방향 IEEE 부동 소수점 인코딩 |
| MQENC_FLOAT_S390 | z/OS® 아키텍처 부동 소수점 인코딩 |
| MQENC_NATIVE | 기본 시스템 인코딩 |

특성 값을 구성하기 위해 애플리케이션은 다음과 같이 세 개의 상수를 추가할 수 있습니다.

- 이름이 MQENC_INTEGER로 시작하는 상수. 2진 정수의 표시를 지정합니다.
- 이름이 MQENC_DECIMAL로 시작하는 상수. 팩형 10진 정수의 표시를 지정합니다.
- 이름이 MQENC_FLOAT로 시작하는 상수. 부동 소수점 수를 지정합니다.

또는 애플리케이션이 특성을 MQENC_NATIVE로 설정할 수 있습니다. 이 값은 환경에 따라 다릅니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_EXCEPTIONMESSAGE*

**데이터 유형:**
문자열

**특성:**
메시지

메시지가 예외 목적지로 송신된 이유를 설명하는 텍스트입니다. 이 특성은 읽기 전용입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

### *JMS_IBM_EXCEPTIONPROBLEMDESTINATION*

**데이터 유형:**
문자열

**특성:**
메시지

메시지가 예외 목적지에 송신되기 전에 있었던 목적지 이름입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

### *JMS_IBM_EXCEPTIONREASON*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지가 예외 목적지에 송신된 이유를 나타내는 이유 코드입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

### *JMS_IBM_EXCEPTIONTIMESTAMP*

**데이터 유형:**
System.Int64

**특성:**
메시지

메시지가 예외 목적지로 전송된 시간입니다.

이 시간은 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

### *JMS_IBM_FEEDBACK*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서 메시지의 속성을 나타내는 코드입니다.

특성의 유효값은 메시지 디스크립터의 **Feedback** 필드에서 지정할 수 있는 피드백 코드와 이유 코드입니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *JMS_IBM_FORMAT*

**데이터 유형:**
문자열

**특성:**
메시지

메시지에서 애플리케이션 데이터의 속성입니다.

특성의 유효값은 메시지 디스크립터의 **Format** 필드에서 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_LAST_MSG_IN_GROUP*

**데이터 유형:**
System.Boolean

**특성:**
메시지

메시지가 메시지 그룹에서 마지막 메시지인지 나타냅니다.

메시지가 메시지 그룹의 마지막 메시지인 경우 이 특성을 true로 설정하십시오. 그렇지 않으면 이 특성을 false로 설정하거나 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

true 값은 메시지 디스크립터의 **MsgFlags** 필드에 지정할 수 있는 상태 플래그 MQMF_LAST_MSG_IN_GROUP에 해당합니다.

이 특성은 발행/구독 도메인에서 무시되며 애플리케이션이 서버 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_MSGTYPE*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지의 유형입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| MQMT_DATAGRAM | 응답을 요청하지 않는 메시지입니다. |
| MQMT_REQUEST | 응답이 필요한 메시지입니다. |
| MQMT_REPLY | 응답 메시지입니다. |
| MQMT_REPORT | 보고 메시지입니다. |

이러한 값은 메시지 디스크립터의 **MsgType** 필드에서 지정할 수 있는 메시지 유형에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_PUTAPPLTYPE*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지를 송신한 애플리케이션의 유형입니다.

특성의 유효값은 메시지 디스크립터의 **PutApplType** 필드에서 지정할 수 있는 애플리케이션 유형입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_PUTDATE*

**데이터 유형:**
문자열

**특성:**
메시지

메시지가 송신된 날짜입니다.

특성의 유효값은 메시지 디스크립터의 **PutDate** 필드에서 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_PUTTIME*

**데이터 유형:**
문자열

**특성:**
메시지

메시지가 송신된 시간입니다.

특성의 유효값은 메시지 디스크립터의 **PutTime** 필드에서 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### *JMS_IBM_REPORT_COA*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '도착 확인' 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| MQRO_COA | 원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 '도착 시 확인' 보고 메시지를 요청합니다. |
| MQRO_COA_WITH_DATA | 원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 '도착 시 확인' 보고 메시지를 요청합니다. |
| MQRO_COA_WITH_FULL_DATA | 원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 '도착 시 확인' 보고 메시지를 요청합니다. |

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다. 이러한 옵션에 대한 자세한 정보는 Report (MQLONG)를 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

### *JMS_IBM_REPORT_COD*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '전달 확인' 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| MQRO_COD | 원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 '전달 시 확인' 보고 메시지를 요청합니다. |
| MQRO_COD_WITH_DATA | 원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 '전달 시 확인' 보고 메시지를 요청합니다. |
| MQRO_COD_WITH_FULL_DATA | 원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 '전달 시 확인' 보고 메시지를 요청합니다. |

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *JMS_IBM_REPORT_DISCARD_MSG*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지를 의도한 목적지에 전달할 수 없을 경우 메시지를 버리는 요청입니다.

메시지를 원하는 목적지에 전달할 수 없는 경우 메시지를 버리도록 요청하려면 특성을 MQRO_DISCARD_MSG로 설정하십시오. 메시지를 데드-레터 큐에 넣거나 예외 목적지로 전송해야 하는 경우 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO_DISCARD_MSG 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

### *JMS_IBM_REPORT_EXCEPTION*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 예외 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| MQRO_EXCEPTION | 원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 예외 보고 메시지를 요청합니다. |
| MQRO_EXCEPTION_WITH_DATA | 원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 예외 보고 메시지를 요청합니다. |
| MQRO_EXCEPTION_WITH_FULL_DATA | 원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 예외 보고 메시지를 요청합니다. |

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *JMS_IBM_REPORT_EXPIRATION*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 만기 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| MQRO_EXPIRATION | 원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 만기 보고 메시지를 요청합니다. |
| MQRO_EXPIRATION_WITH_DATA | 원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 만기 보고 메시지를 요청합니다. |
| MQRO_EXPIRATION_WITH_FULL_DATA | 원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 만기 보고 메시지를 요청합니다. |

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *JMS_IBM_REPORT_NAN*

**데이터 유형:**
System.Int32

**특성:**
메시지

부정적인 조치 보고서 메시지를 요청합니다.

부정적 조치 알림 보고 메시지를 요청하려면 이 특성을 MQRO_NAN으로 설정하십시오. 부정적 조치 알림 보고 메시지가 필요하지 않으면 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO_NAN 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

### *JMS_IBM_REPORT_PAN*

**데이터 유형:**
System.Int32

**특성:**
메시지

긍정적인 조치 알림 보고서 메시지를 요청합니다.

긍적적 조치 알림 보고 메시지를 요청하려면 이 특성을 MQRO_PAN으로 설정하십시오. 긍정적 조치 알림 보고 메시지가 필요하지 않으면 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO_PAN 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

### *JMS_IBM_REPORT_PASS_CORREL_ID*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| MQRO_PASS_CORREL_ID | 모든 보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다. |
| MQRO_COPY_MSG_ID_TO_CORREL_ID | 모든 보고서나 응답 메시지의 상관 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다. |

이러한 값은 메시지 설명자의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

이 특성의 기본값은 MQRO_COPY_MSG_ID_TO_CORREL_ID입니다.

### *JMS_IBM_REPORT_PASS_MSG_ID*

**데이터 유형:**
System.Int32

**특성:**
메시지

보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| MQRO_PASS_MSG_ID | 모든 보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다. |
| MQRO_NEW_MSG_ID | 각 보고서나 응답 메시지에 대해 새 메시지 ID가 생성되도록 요청합니다. |

이러한 값은 메시지 디스크립터의 Report 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

이 특성의 기본값은 MQRO_NEW_MSG_ID입니다.

### *JMS_IBM_RETAIN*

**데이터 유형:**
System.Int32

**특성:**
메시지

이 특성을 설정하면 메시지를 보유된 발행물로 처리하도록 큐 관리자에 표시합니다. 구독자가 토픽에서 메시지를 수신하는 경우, 이전 릴리스에서 수신된 메시지를 넘어서, 구독한 후 즉시 추가 메시지를 수신할 수 있습니다. 이러한 메시지는 구독하는 토픽에 대해 선택적으로 보유된 발행물입니다. 구독과 일치하는 각 토픽에 보유된 발행물이 있는 경우 이 발행물을 구독 중인 메시지 이용자에 전달할 수 있게 됩니다.

이 특성에 유효한 유일한 값은 RETAIN_PUBLICATION입니다. 기본적으로 이 특성은 설정되어 있지 않습니다.

**참고:** 이 특성은 발행/구독 도메인에만 관련됩니다.

### *JMS_IBM_SYSTEM_MESSAGEID*

**데이터 유형:**
문자열

**특성:**
메시지

서비스 통합 버스 내에서 메시지를 고유하는 식별하는 ID입니다. 이 특성은 읽기 전용입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에만 관련됩니다.

### *JMSX_APPID*

**데이터 유형:**
문자열

**특성:**
메시지

메시지를 송신한 애플리케이션의 이름입니다.

이 특성은 JMS 이름이 JMSXAppID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙 버전 *1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

### *JMSX_DELIVERY_COUNT*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지 전달 시도 횟수입니다.

이 특성은 JMS 이름이 JMSXDeliveryCount인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙 버전 *1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

### *JMSX_GROUPID*

**데이터 유형:**
문자열

**특성:**
메시지

메시지가 속한 메시지 그룹의 ID입니다.

이 특성은 JMS 이름이 JMSXGroupID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙 버전 *1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

### *JMSX_GROUPSEQ*

**데이터 유형:**
System.Int32

**특성:**
메시지

메시지 그룹에 있는 메시지의 순서 번호입니다.

이 특성은 JMS 이름이 JMSXGroupSeq인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙 버전 *1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

### *JMSX_USERID*

**데이터 유형:**
문자열

**특성:**
메시지

메시지를 송신한 애플리케이션과 연관된 사용자 ID입니다.

이 특성은 JMS 이름이 JMSXUserID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙 버전 *1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

### *XMSC_ASYNC_EXCEPTIONS*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

이 특성은 연결이 끊기거나 XMS API 호출에 비동기적으로 예외가 발생하는 경우에만 XMS에서 ExceptionListener에 알리는지 여부를 판별합니다. 이 특성은 등록된 ExceptionListener가 있고 이 ConnectionFactory에서 작성된 모든 연결에 적용됩니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_ASYNC_EXCEPTIONS_ALL**
동기 API 호출의 범위 밖에서 비동기적으로 발견된 예외와, 연결이 끊긴 모든 예외가 ExceptionListener로 전송됩니다.

**XMSC_ASYNC_EXCEPTIONS_CONNECTIONBROKEN**
끊긴 연결을 나타내는 예외만 ExceptionListener로 전송됩니다. 비동기 처리 중에 발생한 다른 예외는 ExceptionListener에 보고되지 않으므로 애플리케이션에 이 예외 정보가 제공되지 않습니다.

기본적으로 이 특성은 XMSC_ASYNC_EXCEPTIONS_ALL로 설정됩니다.

### *XMSC_CLIENT_ID*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

연결에 사용하는 클라이언트 ID입니다.

클라이언트 ID는 발행/구독 도메인에서 지속 가능한 구독을 지원하는 데만 사용되며 포인트-투-포인트 도메인에서는 무시됩니다. 클라이언트 ID 설정에 대한 자세한 정보는 21 페이지의 『ConnectionFactories 오브젝트 및 Connection 오브젝트』의 내용을 참조하십시오.

이 특성은 실시간 브로커 연결에는 관련되지 않습니다.

### *XMSC_CONNECTION_TYPE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

애플리케이션이 연결되는 메시징 서버의 유형입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_CT_RTT | 브로커에 대한 실시간 연결입니다. |
| XMSC_CT_WMQ | WebSphere MQ 큐 관리자에 대한 연결입니다. |
| XMSC_CT_WPM | WebSphere 서비스 통합 버스에 대한 연결입니다. |

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_DELIVERY_MODE*

**데이터 유형:**
System.Int32

**특성:**
목적지

**URI에서 사용되는 이름:**

`persistence`(WebSphere MQ 목적지의 경우)
`deliveryMode`(WebSphere 기본 메시징 제공자 목적지의 경우)

목적지에 전송되는 메시지의 전달 모드입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_DELIVERY_NOT_PERSISTENT | 목적지에 전송된 메시지가 비지속적입니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다. 목적지가 WebSphere MQ 큐인 경우 큐 속성인 *DefPersistence* 값도 무시됩니다. |
| XMSC_DELIVERY_PERSISTENT | 목적지에 전송된 메시지가 지속적입니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다. 목적지가 WebSphere MQ 큐인 경우 큐 속성인 *DefPersistence* 값도 무시됩니다. |
| XMSC_DELIVERY_AS_APP | 목적지에 전송된 메시지에 전송 호출에서 지정된 전달 모드가 적용됩니다. 전송 호출이 전달 모드를 지정하지 않으면 메시지 작성자의 기본 전달 모드가 대신 사용됩니다. 목적지가 WebSphere MQ 큐인 경우 큐 속성인 *DefPersistence* 값이 무시됩니다. |
| XMSC_DELIVERY_AS_DEST | 목적지가 WebSphere MQ 큐인 경우 큐에 놓인 메시지에 큐 속성인 *DefPersistence* 값에서 지정된 전달 모드가 적용됩니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다.<br><br>목적지가 WebSphere MQ 큐가 아닌 경우에는 XMSC_DELIVERY_AS_APP의 내용이 동일하게 적용됩니다. |

기본값은 XMSC_DELIVERY_AS_APP입니다.

### *XMSC_IC_PROVIDER_URL*

**데이터 유형:**
문자열

**특성:**
InitialContext

이를 사용하여 JNDI 네이밍 디렉토리를 찾으므로 COS 네이밍 서비스가 웹 서비스와 동일한 서버에 있을 필요가 없습니다.

### *XMSC_IC_SECURITY_AUTHENTICATION*

**데이터 유형:**
문자열

**특성:**
InitialContext

Java 컨텍스트 인터페이스 SECURITY_AUTHENTICATION을 기반으로 합니다. 이 특성은 COS 이름 지정 컨텍스트에만 적용될 수 있습니다.

### *XMSC_IC_SECURITY_CREDENTIALS*

**데이터 유형:**
문자열

**특성:**
InitialContext

Java 컨텍스트 인터페이스 SECURITY_CREDENTIALS를 기반으로 합니다. 이 특성은 COS 이름 지정 컨텍스트에만 적용될 수 있습니다.

### *XMSC_IC_SECURITY_PRINCIPAL*

**데이터 유형:**
문자열

**특성:**
InitialContext

Java 컨텍스트 인터페이스 SECURITY_PRINCIPAL을 기반으로 합니다. 이 특성은 COS 이름 지정 컨텍스트에만 적용될 수 있습니다.

### *XMSC_IC_SECURITY_PROTOCOL*

**데이터 유형:**
문자열

**특성:**
InitialContext

Java 컨텍스트 인터페이스 SECURITY_PROTOCOL을 기반으로 합니다. 이 특성은 COS 네이밍 컨텍스트에만 적용됩니다.

### *XMSC_IC_URL*

**데이터 유형:**
문자열

**특성:**
InitialContext

LDAP 및 FileSystem 컨텍스트의 경우 관리 오브젝트가 들어 있는 저장소의 주소입니다.

COS 네이밍 컨텍스트의 경우, 디렉토리의 오브젝트를 찾는 웹 서비스의 주소입니다.

### *XMSC_IS_SUBSCRIPTION_MULTICAST*

**데이터 유형:**
System.Boolean

**특성:**
MessageConsumer

메시지가 WebSphere MQ 멀티캐스트 전송을 사용하여 메시지 이용자로 전달되는지 여부를 나타냅니다. 이 특성은 읽기 전용입니다.

메시지가 WebSphere MQ 멀티캐스트 전송를 사용하여 메시지 이용자에게 전달되는 경우 이 특성의 값은 true입니다. 그렇지 않은 경우 값은 false입니다.

이 특성은 브로커에 대한 실시간 연결에만 관련됩니다.

### *XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST*

**데이터 유형:**
System.Boolean

**특성:**
MessageConsumer

메시지가 WebSphere MQ 멀티캐스트 전송을 사용하여 신뢰할 수 있는 서비스 품질(QoS)로 메시지 이용자에게 전달되는지 여부를 나타냅니다. 이 특성은 읽기 전용입니다.

메시지가 신뢰할 수 있는 품질로 WebSphere MQ 멀티캐스트 전송를 사용하여 메시지 이용자에게 전달되는 경우 이 특성의 값은 true입니다. 그렇지 않은 경우 값은 false입니다.

이 특성은 브로커에 대한 실시간 연결에만 관련됩니다.

### *XMSC_JMS_MAJOR_VERSION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionMetaData

XMS의 기반이 되는 JMS 스펙의 주 버전 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_JMS_MINOR_VERSION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionMetaData

XMS의 기반이 되는 JMS 스펙의 부 버전 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_JMS_VERSION*

**데이터 유형:**
문자열

**특성:**
ConnectionMetaData

XMS의 기반이 되는 JMS 스펙의 부 버전 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_MAJOR_VERSION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionMetaData

XMS 클라이언트의 버전 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_MINOR_VERSION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionMetaData

XMS 클라이언트의 릴리스 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_PASSWORD*

**데이터 유형:**
바이트 배열

**특성:**
ConnectionFactory

애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 비밀번호입니다. 비밀번호는 XMSC_USERID 특성과 함께 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

분산 플랫폼에서 WebSphere MQ 에 연결하고 연결 팩토리의 XMSC_USERID 특성을 설정하는 경우 로그온한 사용자의 **userid** 와 일치해야 합니다. 이러한 특성을 설정하지 않으면 큐 관리자는 기본적으로 로그인된 사용자의 **userid**를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 WebSphere MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다. 클라이언트 인증 엑시트의 작성에 대한 자세한 정보는 WebSphere MQ 클라이언트 매뉴얼의 인증 주제에서 찾을 수 있습니다.

z/OS에서 WebSphere MQ에 연결할 때 사용자를 인증하려면 보안 종료를 사용해야 합니다.

### *XMSC_PRIORITY*

**데이터 유형:**
System.Int32

**특성:**
목적지

**URI에서 사용되는 이름:**
priority

목적지에 전송되는 메시지의 우선순위입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| 가장 낮은 우선순위는 0이며 가장 높은 우선순위는 9인 정수 범위 | 목적지로 전송되는 메시지에 지정된 우선순위가 있습니다. 메시지 작성자의 기본 우선순위 또는 Send 호출에서 지정된 우선순위는 무시됩니다. 목적지가 WebSphere MQ 큐인 경우 큐 속성 **DefPriority**의 값도 무시됩니다. |
| XMSC_PRIORITY_AS_APP | 목적지로 전송된 메시지에 Send 호출에서 지정된 우선순위가 있습니다. Send 호출이 우선순위를 지정하지 않으면 메시지 작성자의 기본 우선순위가 대신 사용됩니다. 목적지가 WebSphere MQ 큐인 경우 큐 속성 **DefPriority**의 값이 무시됩니다. |
| XMSC_PRIORITY_AS_DEST | 목적지가 WebSphere MQ 큐인 경우 큐에 놓인 메시지에 큐 속성 **DefPriority**의 값에서 지정된 우선순위가 적용됩니다. 메시지 작성자의 기본 우선순위 또는 Send 호출에서 지정된 우선순위는 무시됩니다.<br><br>목적지가 WebSphere MQ 큐가 아닌 경우에는 XMSC_PRIORITY_AS_APP의 내용이 동일하게 적용됩니다. |

기본값은 XMSC_PRIORITY_AS_APP입니다.

WebSphere MQ 실시간 전송 및 WebSphere MQ 멀티캐스트 전송는 메시지의 우선순위에 따라 아무런 조치도 수행하지 않습니다.

### *XMSC_PROVIDER_NAME*

**데이터 유형:**
문자열

**특성:**
ConnectionMetaData

XMS 클라이언트의 제공자입니다. 이 특성은 읽기 전용입니다.

### *XMSC_RTT_BROKER_PING_INTERVAL*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

이 시간 간격(밀리초) 후 XMS .NET에서 활동을 감지하기 위해 실시간 메시징 서버에 대한 연결을 검사합니다. 활동이 감지되지 않으면, 클라이언트는 ping을 초기화합니다. ping에 대한 응답이 감지되지 않으면 연결이 끊어집니다.

이 특성의 기본값은 30000입니다.

### *XMSC_RTT_CONNECTION_PROTOCOL*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

브로커에 대한 실시간 연결에 사용되는 통신 프로토콜입니다.

특성의 값은 XMSC_RTT_CP_TCP여야 합니다. 이 값은 TCP/IP를 통해 브로커에 실시간으로 연결됨을 의미합니다. 기본값은 XMSC_RTT_CP_TCP입니다.

### *XMSC_RTT_HOST_NAME*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

브로커가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.

이 특성은 XMSC_RTT_PORT 특성과 함께 사용하여 브로커를 식별합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_RTT_LOCAL_ADDRESS*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

브로커에 대한 실시간 연결에 사용되는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 특성은 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 유용하며 사용자가 실시간 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나

뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이며 특성이 설정되어 있지 않으면 인터페이스가 임의로 선택됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_RTT_MULTICAST*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**
mulicast

연결 팩토리 또는 목적지의 멀티캐스트 설정입니다. 토픽인 목적지만 이 특성을 보유할 수 있습니다.

애플리케이션은 이 특성을 사용하여 브로커에 대한 실시간 연결과 관련하여 멀티캐스트를 사용할 수 있도록 하며 멀티캐스트가 사용 가능한 경우 멀티캐스트를 사용하여 메시지를 브로커에서 메시지 이용자로 전달하는 방법을 정확하게 지정합니다. 이 특성은 메시지 작성자가 브로커에 메시지를 전송하는 방법에는 영향을 미치지 않습니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_RTT_MULTICAST_DISABLED | 메시지가 WebSphere MQ 멀티캐스트 전송을 사용하여 메시지 처리자에 전달되지 않습니다. 이 값은 ConnectionFactory 오브젝트의 기본값입니다. |
| XMSC_RTT_MULTICAST_ASCF | 메시지가 메시지 이용자와 연관된 연결 팩토리의 멀티캐스트 설정에 따라 메시지 이용자로 전달됩니다. 연결 팩토리의 멀티캐스트 설정은 연결될 때 표시됩니다. 이 값은 Destination 오브젝트에만 유효하며 Destination 오브젝트의 기본값입니다. |
| XMSC_RTT_MULTICAST_ENABLED | 브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 WebSphere MQ 멀티캐스트 전송를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되면 신뢰할 수 있는 서비스 품질(QoS)이 사용됩니다. |
| XMSC_RTT_MULTICAST_RELIABLE | 브로커에서 토픽에 신뢰할 수 있는 멀티캐스트가 구성된 경우, 메시지가 신뢰할 수 있는 서비스 품질(QoS)의 WebSphere MQ 멀티캐스트 전송를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되지 않으면 토픽에 대한 메시지 이용자를 작성할 수 없습니다. |
| XMSC_RTT_MULTICAST_NOT_RELIABLE | 브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 WebSphere MQ 멀티캐스트 전송를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되더라도 신뢰할 수 있는 서비스 품질(QoS)이 사용되지 않습니다. |

### *XMSC_RTT_PORT*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

브로커가 수신 요청을 청취하는 포트 번호입니다. 브로커에서, 이 포트에서 청취하려면 Real-timeInput 또는 Real-timeOptimizedFlow 메시지 처리 노드를 구성해야 합니다.

이 특성은 XMSC_RTT_HOST_NAME 특성과 함께 사용하여 브로커를 식별합니다.

이 특성의 기본값은 XMSC_RTT_DEFAULT_PORT 또는 1506입니다.

### *XMSC_TIME_TO_LIVE*

**데이터 유형:**
System.Int32

**특성:**
목적지

**URI에서 사용되는 이름:**

expiry(WebSphere MQ 목적지에 사용)
timeToLive(WebSphere 기본 메시징 제공자 목적지의 경우)

목적지에 송신되는 메시지의 TTL(Time to Live).

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| 0 | 목적지에 전송되는 메시지가 만기되지 않습니다. |
| 양수 | 목적지로 전송되는 메시지에 밀리초 단위로 지정된 TTL(Time to Live)이 있습니다. 메시지 작성자의 기본 TTL 또는 Send 호출에서 지정된 모든 TTL이 무시됩니다. |
| XMSC_TIME_TO_LIVE_AS_APP | 목적지에 전송되는 메시지에 Send 호출에서 지정된 TTL이 있습니다. Send 호출이 TTL을 지정하지 않으면 메시지 작성자의 기본 TTL이 대신 사용됩니다. |

기본값은 XMSC_TIME_TO_LIVE_AS_APP입니다.

### *XMSC_USERID*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 사용자 ID입니다. 사용자 ID는 XMSC_PASSWORD 특성과 함께 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

WebSphere MQ 분산 플랫폼에 연결하고 연결 팩토리의 XMSC_USERID 특성을 설정하는 경우 로그온한 사용자의 **userid** 와 일치해야 합니다. 이러한 특성을 설정하지 않으면 큐 관리자는 기본적으로 로그인된 사용자의 **userid**를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 WebSphere MQ에 구성된 클라이언트 인증 종료를 작성할 수 있습니다.

z/OS에서 WebSphere MQ에 연결할 때 사용자를 인증하려면 보안 종료를 사용해야 합니다.

### *XMSC_VERSION*

**데이터 유형:**
문자열

**특성:**
ConnectionMetaData

XMS 클라이언트의 버전 ID입니다. 이 특성은 읽기 전용입니다.

### *XMSC_WMQ_BROKER_CONTROLQ*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

브로커가 사용하는 제어 큐의 이름입니다.

이 특성의 기본값은 SYSTEM.BROKER.CONTROL.QUEUE입니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WMQ_BROKER_PUBQ*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 발행하는 메시지를 송신하는 브로커에서 모니터하는 큐의 이름입니다.

이 특성의 기본값은 SYSTEM.BROKER.DEFAULT.STREAM입니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WMQ_BROKER_QMGR*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

브로커가 연결되는 큐 관리자의 이름입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WMQ_BROKER_SUBQ*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

지속 불가능한 메시지 이용자의 구독자 큐 이름입니다.

구독자 큐 이름은 다음 문자로 시작되어야 합니다.

SYSTEM.JMS.ND.

지속 불가능한 모든 이용자가 구독자 큐를 공유하도록 하려면 공유 큐의 완전한 이름을 지정하십시오. 지정된 이름의 큐가 존재해야 애플리케이션이 지속 불가능한 메시지 이용자를 작성할 수 있습니다.

각 비지속 메시지 이용자가 고유한 독점 구독자 큐에서 메시지를 검색하게 하려면 별표(*)로 끝나는 큐 이름을 지정하십시오. 그런 다음 애플리케이션이 지속 불가능한 메시지 이용자를 작성할 때 XMS 클라이언트는 메시지 이용자가 독점적으로 사용할 동적 큐를 작성합니다. XMS 클라이언트는 이 특성 값을 사용하여 동적 큐를 작성하는 데 사용되는 오브젝트 디스크립터에 있는 **DynamicQName** 필드의 컨텐츠를 설정합니다.

이 특성의 기본값은 SYSTEM.JMS.ND.SUBSCRIBER.QUEUE입니다. 이 값은 XMS가 기본적으로 공유된 큐 접근 방법을 사용한다는 의미입니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WMQ_BROKER_VERSION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**
brokerVersion

애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다. 토픽인 목적지만 이 특성을 보유할 수 있습니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WMQ_BROKER_V1 | 애플리케이션이 WebSphere MQ 발행/구독 브로커를 사용하고 있습니다.<br><br>WebSphere MQ 발행/구독에서 WebSphere Message Broker로 마이그레이션했지만, 해당 애플리케이션을 변경하지 않은 경우 애플리케이션이 이 값을 사용할 수 있습니다. |
| XMSC_WMQ_BROKER_V2 | 애플리케이션이 IBM Integration Bus의 브로커를 사용합니다. |
| XMSC_WMQ_BROKER_UNSPECIFIED | 브로커를 마이그레이션한 후 RFH2 헤더가 더 이상 사용되지 않도록 이 특성을 설정하십시오. 마이그레이션하고 나면 이 특성은 더 이상 적절하지 않습니다. |

connectionfactory의 기본값은 XMSC_WMQ_BROKER_UNSPECIFIED이지만 기본적으로 이 특성은 목적지에 대해서는 설정되지 않습니다. 목적지에 대한 특성을 설정하면 연결 팩토리 특성에서 지정한 값을 모두 대체합니다.

### *XMSC_WMQ_CCDTURL*

**데이터 유형:**
System.String

**특성:**
ConnectionFactory

클라이언트 채널 정의 테이블을 포함하는 파일의 이름과 위치를 식별하고 이 파일의 액세스 방법을 지정하는 URL(Uniform Resource Locator)입니다.

기본적으로 이 특성은 설정되어 있지 않습니다.

### *XMSC_WMQ_CCSID*

**데이터 유형:**
System.Int32

**특성:**
목적지

**URI에서 사용되는 이름:**
CCSID

XMS 클라이언트에서 메시지를 목적지로 전달할 때 메시지 본문에 있는 문자 데이터의 문자열에 사용되는 코드화된 문자 세트 ID(CCSID) 또는 코드 페이지입니다. 개별 메시지에 대해 설정된 경우 JMS_IBM_CHARACTER_SET 특성이 이 특성에서 목적지에 대해 지정한 CCSID를 대체합니다.

이 특성의 기본값은 1208입니다.

이 특성은 목적지에서 수신한 메시지가 아니라 목적지로 전송된 메시지에만 관련됩니다.

### *XMSC_WMQ_CHANNEL*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

연결에 사용되는 채널 이름입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

### *XMSC_WMQ_CLIENT_RECONNECT_OPTIONS*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

이 특성은 이 팩토리에서 작성한 새 연결에 대한 클라이언트 재연결 옵션을 지정합니다. 이는 XMSC에 있으며 다음 중 하나입니다.

- WMQ_CLIENT_RECONNECT_AS_DEF(기본값). `mqclient.ini` 파일에 지정된 값을 사용하십시오. 채널 스탠자 내의 **DefRecon** 특성을 사용하여 값을 설정하십시오. 다음 중 하나로 설정할 수 있습니다.
  1. 예 WMQ_CLIENT_RECONNECT 옵션으로 사용
  2. 아닙니다. 기본값: 다시 연결 옵션을 지정하지 않음
  3. QMGR. WMQ_CLIENT_RECONNECT_Q_MGR 옵션으로 사용
  4. DISABLED. WMQ_CLIENT_RECONNECT_DISABLED 옵션으로 사용
- WMQ_CLIENT_RECONNECT. 연결 이름 목록에 지정된 큐 관리자에 다시 연결합니다.
- WMQ_CLIENT_RECONNECT_Q_MGR. 원래 연결된 동일한 큐 관리자에 다시 연결합니다. 연결하려고 하는 큐 관리자(연결 이름 목록에서 지정됨)에 원래 연결되는 큐 관리자와 다른 QMID가 있는 경우 `MQRC_RECONNECT_QMID_MISMATCH`가 리턴됩니다.
- WMQ_CLIENT_RECONNECT_DISABLED. 다시 연결을 사용할 수 없습니다.

### *XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT 특성은 관리 XMS .NET 클라이언트에만 유효합니다.

이 특성은 클라이언트 연결이 다시 연결하려고 시도하는 시간(초)을 지정합니다.

이 시간 동안 다시 연결하려고 시도한 후에는 클라이언트가 MQRC_RECONNECT_FAILED와 함께 실패합니다. 이 특성의 기본 설정은 XMSC.WMQ_CLIENT_RECONNECT_TIMEOUT_DEFAULT입니다.

이 특성의 기본값은 1800입니다.

### *XMSC_WMQ_CONNECTION_MODE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

애플리케이션이 큐 관리자에 연결할 때 사용되는 모드입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WMQ_CM_BINDINGS | 최적의 성능을 위한 바인딩 모드의 큐 관리자에 대한 연결입니다. 이 값은 C/C++에 대한 기본값입니다. |
| XMSC_WMQ_CM_CLIENT | 완전한 관리 스택을 위한 클라이언트 모드의 큐 관리자에 대한 연결입니다. 이 값은 .NET에 대한 기본값입니다. |
| XMSC_WMQ_CM_CLIENT_UNMANAGED(.NET만 해당) | 비관리 클라이언트 스택을 강제 실행하는 큐 관리자에 대한 연결입니다. |

**관련 개념**

.NET의 관리되는 운영 및 관리되지 않는 운영

관리 코드는 .NET 공통 언어 런타임 환경 내에서 배타적으로 실행되며 전적으로 해당 런타임에서 제공되는 서비스에 종속됩니다. 애플리케이션의 일부가 .NET 공용 언어 런타임 환경 외부에서 실행되거나 서비스를 호출하면 애플리케이션은 비관리로 분류됩니다.

### *XMSC_WMQ_CONNECTION_NAME_LIST*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

이 특성은 해당 연결이 끊어진 이후 클라이언트가 재연결을 시도하는 호스트를 지정합니다.

연결 이름 목록은 쉼표로 구분된 호스트/IP 포트 쌍의 목록입니다. 이 특성의 기본 설정은 WMQ_CONNECTION_NAME_LIST_DEFAULT입니다.

예를 들어 `127.0.0.1(1414),host2.example.com(1400)`입니다.

이 특성의 기본 설정은 `localhost(1414)`입니다.

### *XMSC_WMQ_DUR_SUBQ*

**데이터 유형:**
문자열

**특성:**
목적지

목적지에서 메시지를 수신하는 지속 가능한 구독자의 구독자 큐 이름입니다. 토픽인 목적지만 이 특성을 보유할 수 있습니다.

구독자 큐 이름은 다음 문자로 시작되어야 합니다.

```
SYSTEM.JMS.D.
```

지속 가능한 모든 구독자가 구독자 큐를 공유하도록 하려면 공유 큐의 완전한 이름을 지정하십시오. 지정된 이름의 큐가 존재해야 애플리케이션이 지속 가능한 구독자를 작성할 수 있습니다.

지속 가능한 각 구독자가 고유한 독점 구독자 큐에서 메시지를 검색하게 하려면 별표(*)로 끝나는 큐 이름을 지정하십시오. 그런 다음 애플리케이션이 지속 가능한 구독자를 작성할 때 XMS 클라이언트는 지속 가능한 구독자가 독점 사용할 동적 큐를 작성합니다. XMS 클라이언트는 이 특성 값을 사용하여 동적 큐를 작성하는 데 사용되는 오브젝트 디스크립터에 있는 **DynamicQName** 필드의 컨텐츠를 설정합니다.

이 특성의 기본값은 SYSTEM.JMS.D.SUBSCRIBER.QUEUE입니다. 이 값은 XMS가 기본적으로 공유된 큐 접근 방법을 사용한다는 의미입니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WMQ_ENCODING*

**데이터 유형:**
System.Int32

**특성:**
목적지

XMS 클라이언트가 메시지를 목적지에 전달할 때 메시지 본문의 숫자 데이터를 표시하는 방법입니다. 개별 메시지에 설정된 경우 JMS_IBM_ENCODING 특성이 이 특성에서 목적지에 대해 지정한 인코딩을 대체합니다. 이 특성은 2진 정수, 팩형 10진수 정수 및 부동 소수점 숫자의 표시를 지정합니다.

특성의 올바른 값은 메시지 설명자의 **Encoding** 필드에 지정할 수 있는 값과 동일합니다.

애플리케이션은 다음과 같은 이름 지정된 상수를 사용하여 특성을 설정할 수도 있습니다.

| 이름 지정된 상수 | 의미 |
|---|---|
| MQENC_INTEGER_NORMAL | 표준 정수 인코딩 |
| MQENC_INTEGER_REVERSED | 역방향 정수 인코딩 |
| MQENC_DECIMAL_NORMAL | 표준 팩형 10진수 인코딩 |
| MQENC_DECIMAL_REVERSED | 역방향 팩형 10진수 인코딩 |
| MQENC_FLOAT_IEEE_NORMAL | 표준 IEEE 부동 소수점 인코딩 |
| MQENC_FLOAT_IEEE_REVERSED | 역방향 IEEE 부동 소수점 인코딩 |
| MQENC_FLOAT_S390 | z/OS 아키텍처 부동 소수점 인코딩 |
| MQENC_NATIVE | 기본 시스템 인코딩 |

특성 값을 구성하기 위해 애플리케이션은 다음과 같이 세 개의 상수를 추가할 수 있습니다.

- 이름이 MQENC_INTEGER로 시작하는 상수. 2진 정수의 표시를 지정합니다.
- 이름이 MQENC_DECIMAL로 시작하는 상수. 팩형 10진 정수의 표시를 지정합니다.
- 이름이 MQENC_FLOAT로 시작하는 상수. 부동 소수점 수를 지정합니다.

또는 애플리케이션이 특성을 MQENC_NATIVE로 설정할 수 있습니다. 이 값은 환경에 따라 다릅니다.

이 특성의 기본값은 MQENC_NATIVE입니다.

이 특성은 목적지에서 수신한 메시지가 아니라 목적지로 전송된 메시지에만 관련됩니다.

### *XMSC_WMQ_FAIL_IF_QUIESCE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**
failIfQuiesce

애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| XMSC_WMQ_FIQ_YES | 큐 관리자가 정지(quiescing) 상태에 있는 경우 특정 메소드에 대한 호출이 실패합니다. 애플리케이션은 큐 관리자가 정지 상태임을 감지하면 즉시 태스크를 완료하고 연결을 닫아 큐 관리자를 중지할 수 있습니다. |

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WMQ_FIQ_NO | 큐 관리자가 정지 상태이기 때문에 메소드에 대한 호출이 실패하지 않습니다. 이 값을 지정하면 애플리케이션이 큐 관리자가 정지 상태임을 감지할 수 없습니다. 애플리케이션은 큐 관리자에 대한 조작을 계속 수행합니다. 따라서 큐 관리자가 중지하는 것을 방지할 수 있습니다. |

연결 팩토리의 기본값은 XMSC_WMQ_FIQ_YES이지만 기본적으로 이 특성은 목적지에 대해서는 설정되지 않습니다. 목적지에 대한 특성을 설정하면 연결 팩토리 특성에서 지정한 값을 모두 대체합니다.

### *XMSC_WMQ_MESSAGE_BODY*

**데이터 유형:**
System.Int32

**특성:**
목적지

이 특성은 XMS 애플리케이션이 메시지 페이로드의 일부(즉, 메시지 본문의 일부)로서 IBM WebSphere MQ 메시지의 MQRFH2를 처리하는지 여부를 판별합니다.

**참고:** 메시지를 목적지로 전송할 때, XMSC_WMQ_MESSAGE_BODY 특성은 기존 XMS 목적지 특성 XMSC_WMQ_TARGET_CLIENT를 대신합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_MESSAGE_BODY_JMS**
**Receive:** 인바운드 XMS 메시지 유형 및 본문은 수신된 IBM WebSphere MQ 메시지에 있는 MQRFH2(있는 경우) 또는 MQMD(MQRFH2가 없는 경우)의 컨텐츠에 의해 판별됩니다.

**전송:** 아웃바운드 XMS 메시지 본문에 XMS 메시지 특성과 헤더 필드를 기반으로 첨부되고 자동으로 생성된 MQRFH2 헤더를 포함합니다.

**XMSC_WMQ_MESSAGE_BODY_MQ**
**Receive:** 수신된 IBM WebSphere MQ 메시지의 컨텐츠 또는 수신된 MQMD의 형식 필드에 상관없이 인바운드 XMS 메시지 유형은 항상 ByteMessage입니다. XMS 메시지 본문은 기본 메시징 제공자 API 호출로 리턴된 변경되지 않은 메시지 데이터입니다. 메시지 본문에 있는 데이터의 문자 세트와 인코딩은 MQMD의 CodedCharSetId 및 Encoding 필드에 의해 판별됩니다. 메시지 본문에 있는 데이터의 형식은 MQMD의 Format 필드에 의해 판별됩니다.

**Send:** 아웃바운드 XMS 메시지 본문은 애플리케이션 페이로드를 있는 그대로 포함하고, 자동 생성된 IBM WebSphere MQ 헤더가 본문에 추가되지 않습니다.

**XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED**
**Receive:** XMS 클라이언트가 이 특성에 적합한 값을 결정합니다. 수신 경로에서 이 값은 WMQ_MESSAGE_BODY_JMS 특성 값입니다.

**Send:** XMS 클라이언트가 이 특성에 적합한 값을 결정합니다. 송신 경로에서 이 값은 XMSC_WMQ_TARGET_CLIENT 특성 값입니다.

기본적으로 이 특성은 XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED로 설정됩니다.

### *XMSC_WMQ_MQMD_MESSAGE_CONTEXT*

**데이터 유형:**
System.Int32

**특성:**
목적지

XMS 애플리케이션이 설정할 메시지 컨텍스트의 레벨을 결정합니다. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_MDCTX_DEFAULT**
아웃바운드 메시지의 경우, MQOPEN API 호출과 MQPMO 구조는 명시적인 메시지 컨텍스트 옵션을 지정하지 않습니다.

**XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT**
MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO_SET_IDENTITY_CONTEXT를 지정하고 MQPMO 구조가 MQPMO_SET_IDENTITY_CONTEXT를 지정합니다.

**XMSC_WMQ_MDCTX_SET_ALL_CONTEXT**
MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO_SET_ALL_CONTEXT를 지정하고 MQPMO 구조가 MQPMO_SET_ALL_CONTEXT를 지정합니다.

기본적으로 이 특성은 XMSC_WMQ_MDCTX_DEFAULT로 설정됩니다.

**참고:** 이 특성은 애플리케이션이 서비스 통합 버스에 연결된 경우에는 관련이 없습니다.

메시지를 보낼 때 다음 특성이 적용되도록 하려면 XMSC_WMQ_MQMD_MESSAGE_CONTEXT 특성을 XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT 특성 값 또는 XMSC_WMQ_MDCTX_SET_ALL_CONTEXT 특성 값으로 설정해야 합니다.

- JMS_IBM_MQMD_USERIDENTIFIER
- JMS_IBM_MQMD_ACCOUNTINGTOKEN
- JMS_IBM_MQMD_APPLIDENTITYDATA

메시지를 보낼 때 다음 특성이 적용되도록 하려면 XMSC_WMQ_MQMD_MESSAGE_CONTEXT 특성을 XMSC_WMQ_MDCTX_SET_ALL_CONTEXT 특성 값으로 설정해야 합니다.

- 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*

**데이터 유형:**
System.Int32

**특성:**
목적지

이 특성은 XMS 애플리케이션이 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_READ_ENABLED_NO**
메시지를 송신할 때 송신된 메시지의 JMS_IBM_MQMD* 특성이 MQMD에서 업데이트된 필드 값을 반영하도록 업데이트되지 않습니다.

메시지를 수신할 때 송신자가 일부 또는 모든 특성을 설정했더라도 JMS_IBM_MQMD* 특성을 수신 메시지에서 사용할 수 없습니다.

**XMSC_WMQ_READ_ENABLED_YES**
메시지를 송신할 때 송신자가 명시적으로 설정하지 않은 해당 특성을 포함하여 MQMD에서 업데이트된 필드 값을 반영하기 위해 송신된 메시지의 모든 JMS_IBM_MQMD* 특성이 업데이트됩니다.

메시지를 수신할 때 송신자가 명시적으로 설정하지 않은 해당 특성을 포함하여 모든 JMS_IBM_MQMD* 특성을 수신 메시지에서 사용할 수 있습니다.

기본적으로 이 특성은 XMSC_WMQ_READ_ENABLED_NO로 설정됩니다.

### *XMSC_WMQ_MQMD_WRITE_ENABLED*

**데이터 유형:**
System.Int32

**특성:**
목적지

이 특성은 XMS 애플리케이션이 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_WRITE_ENABLED_NO**
모든 JMS_IBM_MQMD* 특성이 무시되며, 해당 값이 기본 MQMD 구조로 복사되지 않습니다.

**XMSC_WMQ_WRITE_ENABLED_YES**
JMS_IBM_MQMD* 특성이 처리됩니다. 해당 값이 기본 MQMD 구조로 복사됩니다.

기본적으로 이 특성은 XMSC_WMQ_WRITE_ENABLED_NO로 설정됩니다.

### *XMSC_WMQ_PUT_ASYNC_ALLOWED*

**데이터 유형:**
System.Int32

**특성:**
목적지

이 특성은 메시지 생성자가 비동기 Put을 사용하여 메시지를 이 목적지로 송신할 수 있는지 여부를 판별합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST**
큐 정의 또는 토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_Q_DEF**
큐 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_TOPIC_DEF**
토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC_WMQ _PUT_ASYNC_ALLOWED_DISABLED**
비동기 Put이 허용되지 않습니다.

**XMSC_WMQ _PUT_ASYNC_ALLOWED_ENABLED**
비동기 Put이 허용됩니다.

기본적으로 이 특성은 XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST로 설정됩니다.

**참고:** 이 특성은 애플리케이션이 서비스 통합 버스에 연결하는 경우에는 관련이 없습니다.

### *XMSC_WMQ_READ_AHEAD_ALLOWED*

**데이터 유형:**
System.Int32

**특성:**
목적지

이 특성은 메시지를 수신하기 전에 메시지 이용자와 큐 브라우저가 이 목적지에서 내부 버퍼로 트랜잭트되지 않은 비지속적 메시지를 가져오기 위해 미리 읽기를 사용할 수 있는지 여부를 판별합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF**
큐 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

**XMSC_WMQ_READ_AHEAD_ALLOWED_AS_ TOPIC _DEF**
토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

**XMSC_WMQ_READ_AHEAD_ALLOWED_AS_DEST**
큐 정의 또는 토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

**XMSC_WMQ_READ_AHEAD_ALLOWED_DISABLED**
메시지를 이용하거나 찾아보는 동안 미리 읽기가 허용되지 않습니다.

**XMSC_WMQ_READ_AHEAD_ALLOWED_ENABLED**
미리 읽기가 허용됩니다.

기본적으로 이 특성은 XMSC_WMQ _READ_AHEAD_ALLOWED_AS_DEST로 설정됩니다.

## *XMSC_WMQ_READ_AHEAD_CLOSE_POLICY*

**데이터 유형:**
System.Int32

**특성:**
목적지

비동기 메시지 리스너로 전달되는 메시지의 경우, 메시지 이용자를 닫을 때 내부 미리 읽기 버퍼에서 메시지에 일어나는 사항을 이 특성에서 판별합니다.

이 특성은 목적지에서 메시지를 이용할 때 큐 닫기 옵션을 지정하는 데 사용되지만 목적지로 메시지를 보내는 경우에는 적용되지 않습니다.

찾아보는 중에는 메시지가 큐에서 사용 가능한 상태이므로 이 특성은 큐 브라우저에 대해 무시됩니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT**
리턴하기 전에 현재 메시지 리스너 호출만 완료하고, 내부 미리 읽기 버퍼에 메시지를 남길 수 있습니다. 이 메시지는 이후 제거됩니다.

**XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_ALL**
리턴하기 전에 내부 미리 읽기 버퍼의 모든 메시지가 애플리케이션 메시지 리스너로 전달됩니다.

기본적으로 이 특성은 XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT로 설정됩니다.

**참고:**

- **비정상 애플리케이션 종료**

  XMS 애플리케이션이 갑자기 종료되면 미리 읽기 버퍼에 있는 모든 메시지가 손실됩니다.
- **트랜잭션의 영향**

  애플리케이션이 트랜잭션을 사용할 때 미리 읽기가 사용 안 함으로 설정됩니다. 따라서 애플리케이션이 트랜잭션된 세션을 사용할 때 동작에서 차이가 없습니다.
- **세션 수신확인 모드의 영향**

  수신확인 모드가 XMSC_AUTO_ACKNOWLEDGE 또는 XMSC_DUPS_OK_ACKNOWLEDGE인 경우 미리 읽기가 트랜잭트되지 않은 세션에서 사용할 수 있도록 설정됩니다. 세션의 트랜잭션 여부에 관계없이, 세션 수신확인 모드가 XMSC_CLIENT_ACKNOWLEDGE인 경우 미리 읽기가 사용 안 함으로 설정됩니다.
- **큐 브라우저 및 큐 브라우저 선택자의 영향**

  XMS 애플리케이션에 사용되는 큐 브라우저 및 큐 브라우저 선택자는 미리 읽기 기능을 성능 이점으로 활용합니다. 큐 브라우저를 닫아도 메시지가 추가 작업을 위해 계속 큐에서 사용 가능 상태이므로 성능이 저하되지 않습니다. 미리 읽기의 성능 이점을 제외하고 큐 브라우저와 큐 브라우저 선택자에 미치는 다른 영향이 없습니다.

### *XMSC_WMQ_HOST_NAME*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

큐 관리자가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 사용됩니다. 이 특성은 XMSC_WMQ_PORT 특성과 함께 사용하여 큐 관리자를 식별합니다.

이 특성의 기본값은 `localhost`입니다.

### *XMSC_WMQ_LOCAL_ADDRESS*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

큐 관리자에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.

이 특성의 값은 다음 형식의 문자열입니다.

[*host_name*][(*low_port*)[,*high_port*])]

변수의 의미는 다음과 같습니다.

***host_name***
연결에 사용하는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 정보는 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 필요하며 사용자가 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이고 사용해야 할 인터페이스를 지정하지 않으면 인터페이스가 임의로 선택됩니다.

***low_port***
연결에 사용되는 로컬 포트의 번호입니다.

*high_port*도 함께 지정하면 *low_port*가 포트 번호 범위 중에서 가장 낮은 번호로 해석됩니다.

***high_port***
포트 번호 범위에서 가장 높은 포트 번호입니다. 지정된 범위의 포트 중 하나를 연결에 사용해야 합니다.

문자열의 최대 길이는 48자입니다.

다음은 특성의 올바른 값에 대한 예입니다.

JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

### *XMSC_WMQ_MESSAGE_SELECTION*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

메시지 선택이 XMS 클라이언트 또는 브로커에서 수행되는지를 판별합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WMQ_MSEL_CLIENT | XMS 클라이언트에서 메시지를 선택합니다. |
| XMSC_WMQ_MSEL_BROKER | 브로커를 통해 메시지가 선택됩니다. |

기본값은 XMSC_WMQ_MSEL_CLIENT입니다.

이 특성은 발행/구독 도메인에서만 관련됩니다. XMSC_WMQ_BROKER_VERSION 특성이 XMSC_WMQ_BROKER_V1로 설정된 경우 브로커의 메시지 선택이 지원되지 않습니다.

### *XMSC_WMQ_MSG_BATCH_SIZE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

비동기 메시지 전달을 사용할 때 하나의 일괄처리로 큐에서 검색되는 최대 메시지 수입니다.

애플리케이션이 비동기 메시지 전달을 사용하는 경우 특정 조건에서는 XMS 클라이언트가 각 메시지를 개별적으로 애플리케이션에 전달하기 전에 큐에서 메시지 배치를 검색합니다. 이 특성은 배치에 있을 수 있는 최대 메시지 수를 지정합니다.

특성 값은 양의 정수이며 기본값은 10입니다. 해결해야 하는 특정한 성능 문제점이 있는 경우에만 특성을 다른 값으로 설정하십시오.

애플리케이션이 네트워크를 통해 큐 관리자에 연결된 경우 이 특성의 값을 높이면 네트워크 오버헤드 및 응답 시간을 감소시킬 수 있습니다. 그러나 클라이언트 시스템에 메시지를 저장하는 데 필요한 메모리 양이 증가합니다. 반대로 이 특성의 값을 낮추면 네트워크 오버헤드 및 응답 시간이 증가하지만 메시지를 저장하는 데 필요한 메모리의 양을 감소시킬 수 있습니다.

### *XMSC_WMQ_POLLING_INTERVAL*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

세션 내에 있는 각 메시지 리스너의 큐에 적합한 메시지가 없는 경우, 이 값은 각 메시지 리스너가 해당 큐에서 메시지 가져오기를 다시 시도하기 전에 경과하는 최대 간격(밀리초)입니다.

세션의 메시지 리스너에 사용 가능한 적합한 메시지가 없는 경우가 자주 발생하면 이 특성 값을 높여 보십시오.

특성 값은 양의 정수입니다. 기본값은 5000입니다.

### *XMSC_WMQ_PORT*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

큐 관리자 리스너가 수신 요청을 청취하는 포트 번호입니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 사용됩니다. 이 특성은 XMSC_WMQ_HOST_NAME 특성과 함께 사용하여 큐 관리자를 식별합니다.

이 특성의 기본값은 XMSC_WMQ_DEFAULT_CLIENT_PORT 또는 1414입니다.

### *XMSC_WMQ_PROVIDER_VERSION*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 연결하려고 하는 큐 관리자의 버전, 릴리스, 수정 레벨, 수정팩입니다. 이 특성의 올바른 값은 다음과 같습니다.

- 지정되지 않음

또는 다음 형식 중 하나의 문자열

- V.R.M.F
- V.R.M
- V.R
- V

여기서 V, R, M 및 F는 0 이상의 정수 값입니다.

7 이상의 값은 IBM WebSphere MQ 버전 7.0 큐 관리자에 연결하기 위해 이 버전을 IBM WebSphere MQ 버전 7.0 ConnectionFactory로 사용함을 나타냅니다. 7 미만의 값(예: "6.0.2.0")은 버전 7.0 이하의 큐 관리자에 사용하기 위한 것임을 나타냅니다. 기본값인 unspecified를 사용하면 큐 관리자의 기능을 바탕으로 사용할 수 있는 적용 가능한 특성 및 기능을 판별하여 모든 레벨의 큐 관리자에 연결할 수 있습니다.

기본적으로 이 특성은 "unspecified"로 설정됩니다.

**참고:**

- XMSC_WMQ_PROVIDER_VERSION이 6. 2로 설정되면 소켓 공유가 발생하지 않습니다.
- XMSC_WMQ_PROVIDER_VERSION이 7로 설정되고 채널에 대한 서버 SHARECNV가 0으로 설정된 경우 연결이 실패합니다.
- XMSC_WMQ_PROVIDER_VERSION이 UNSPECIFIED로 설정되고 SHARECNV가 0으로 설정된 경우 IBM WebSphere MQ 버전 7.0 특정 기능을 사용할 수 없습니다.

IBM WebSphere MQ 클라이언트의 버전은 XMS 클라이언트 애플리케이션이 IBM WebSphere MQ 버전 7.0 특정 기능을 사용할 수 있는지 여부에 중요한 역할을 합니다. 다음 표에서는 동작을 설명합니다.

**참고:** 시스템 특성 XMSC_WMQ_OVERRIDEPROVIDERVERSION은 XMSC_WMQ_PROVIDER_VERSION 특성을 대체합니다. 이 특성은 연결 팩토리 설정을 변경할 수 없는 경우에 사용할 수 있습니다.

표 *34. XMS* 클라이언트 - *IBM WebSphere MQ* 버전 *7.0* 특정 기능 사용 가능

| # | XMSC_WMQ_PROVIDER_VERSION | IBM WebSphere MQ 클라이언트 버전 | IBM WebSphere MQ 버전 7.0 기능 |
|---|---|---|---|
| 1 | 지정되지 않음 | 7 | 켜짐(ON) |
| 2 | 지정되지 않음 | 6 | 꺼짐(OFF) |
| 3 | 7 | 7 | 켜짐(ON) |
| 4 | 7 | 6 | 예외 |
| 5 | 6 | 6 | 꺼짐(OFF) |
| 6 | 6 | 7 | 꺼짐(OFF) |

### *XMSC_WMQ_PUB_ACK_INTERVAL*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

XMS 클라이언트가 브로커에서 수신확인을 요청하기 전에 공개자가 공개한 메시지 수입니다.

이 특성의 값을 낮추면 클라이언트가 수신확인을 더 빈번하게 요청하므로 발행자의 성능이 저하됩니다. 값을 올리는 경우 브로커가 실패하면 클라이언트가 예외를 처리하는 데 더 많은 시간이 걸립니다.

특성 값은 양의 정수입니다. 기본 값은 25입니다.

### *XMSC_WMQ_QMGR_CCSID*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

XMS 클라이언트와 WebSphere MQ 클라이언트 사이에서 교환되는 MQI(Message Queue Interface)에서 정의된 문자 데이터의 필드가 있는 코딩된 문자-세트의 ID(CCSID) 또는 코드 페이지입니다. 이 특성은 메시지 본문의 문자 데이터의 문자열에는 적용되지 않습니다.

XMS 애플리케이션 클라이언트 모드에서 큐 관리자에 연결될 때 XMS 클라이언트가 WebSphere MQ 클라이언트에 연결됩니다. 두 클라이언트 사이에서 교환되는 정보에는 MQI에서 정의된 문자 데이터의 필드가 포함됩니다. 정상적인 상황에서는 WebSphere MQ 클라이언트는 이러한 필드가 클라이언트가 실행 중인 시스템의 코드 페이지에 있다고 가정합니다. XMS 클라이언트가 다른 코드 페이지에 있는 이러한 필드를 제공하거나 수신할 것으로 예상되는 경우에는 WebSphere MQ 클라이언트에 알리도록 이 특성을 설정해야 합니다.

WebSphere MQ 클라이언트가 문자 데이터의 해당 필드를 큐 관리자에 전달하는 경우 필요하면 큐 관리자가 사용하는 코드 페이지로 필드 내의 데이터를 변환해야 합니다. 이와 유사하게 WebSphere MQ 클라이언트가 큐 관리자로부터 해당 빌드를 수신하는 경우 필요하면 XMS 클라이언트가 데이터를 수신할 것으로 예상하는 코드 페이지로 필드 내의 데이터를 변환해야 합니다. WebSphere MQ 클라이언트는 이 특성을 사용하여 이러한 데이터 변환을 수행합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성을 설정하는 것은 기본 WebSphere MQ 클라이언트 애플리케이션을 지원하는 WebSphere MQ 클라이언트에 대해 MQCCSID 환경 변수를 설정하는 것과 동일합니다. 이 환경 변수에 대한 자세한 정보는 *WebSphere MQ* 클라이언트를 참조하십시오.

### *XMSC_WMQ_QUEUE_MANAGER*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

연결된 큐 관리자의 이름.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WMQ_RECEIVE_CCSID*

큐 관리자 메시지 변환용 대상 CCSID를 설정하는 목적지 특성입니다. XMSC_WMQ_RECEIVE_CONVERSION을 WMQ_RECEIVE_CONVERSION_QMGR로 설정하지 않은 경우, 값이 무시됩니다.

**데이터 유형:**
정수

**가치:**
임의의 양의 정수입니다.

기본값은 1208입니다.

메시지에 GMO_CONVERT 값을 지정하는 것은 선택사항입니다. GMO_CONVERT 값을 지정한 경우 지정된 값에 따라 변환이 수행됩니다.

### *XMSC_WMQ_RECEIVE_CONVERSION*

큐 관리자가 데이터 변환을 수행할지 판별하는 목적지 특성입니다.

**데이터 유형:**
정수

**값:**
XMSC_WMQ_RECEIVE_CONVERSION_CLIENT_MSG (DEFAULT): XMS 클라이언트에서만 데이터 변환을 수행합니다. 변환은 항상 코드 페이지 1208을 사용하여 수행됩니다.

XMSC_WMQ_RECEIVE_CONVERSION_QMGR: 메시지를 XMS 클라이언트로 송신하기 전에 큐 관리자에서 데이터 변환을 수행합니다.

### *XMSC_WMQ_RECEIVE_EXIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

실행할 채널 수신 종료를 식별합니다.

이 특성의 값은 채널 수신 엑시트를 식별하는 문자열입니다. 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.
- entryPointName은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### *XMSC_WMQ_RECEIVE_EXIT_INIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

호출 시 채널 수신 종료에 전달되는 사용자 데이터입니다.

이 특성의 값은 문자열입니다. 기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 212 페이지의 『XMSC_WMQ_RECEIVE_EXIT』 특성이 설정된 경우에만 관련됩니다.

### *XMSC_WMQ_RESOLVED_QUEUE_MANAGER*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

이 특성을 사용하여 연결된 큐 관리자의 이름을 가져올 수 있습니다.

CCDT(클라이언트 채널 정의 표)에서 사용하는 경우 이 이름은 연결 팩토리에 지정된 큐 관리자 이름과 다를 수 있습니다.

### *XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

이 특성은 연결 이후에 큐 관리자의 ID로 채워집니다.

### *XMSC_WMQ_SECURITY_EXIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

채널 보안 엑시트를 식별합니다.

이 특성의 값은 채널 보안 엑시트를 식별하는 문자열입니다. 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.
- entryPointName은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

문자열의 최대 길이는 128자입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### *XMSC_WMQ_SECURITY_EXIT_INIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

호출 시 채널 보안 엑시트로 전달되는 사용자 데이터입니다.

사용자 데이터 문자열의 최대 길이는 32자입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 213 페이지의 『XMSC_WMQ_SECURITY_EXIT』 특성이 설정된 경우에만 관련됩니다.

### *XMSC_WMQ_SEND_EXIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

채널 전송 종료를 식별합니다.

이 특성의 값은 문자열입니다. 채널 송신 엑시트의 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.
- entryPointName은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### *XMSC_WMQ_SEND_EXIT_INIT*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

호출 시 채널 송신 엑시트로 전달되는 사용자 데이터입니다.

이 특성 값은 쉼표로 구분된 하나 이상의 사용자 데이터 항목의 문자열입니다. 기본적으로 특성이 설정되어 있지 않습니다.

채널 송신 엑시트의 순서에 전달되는 사용자 데이터를 지정하는 규칙은 채널 수신 엑시트의 순서에 전달되는 사용자 데이터를 지정하는 규칙과 동일합니다. 따라서 해당 규칙에 대해서는 212 페이지의 『XMSC_WMQ_RECEIVE_EXIT_INIT』의 내용을 참조하십시오.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 213 페이지의 『XMSC_WMQ_SEND_EXIT』 특성이 설정된 경우에만 관련됩니다.

### *XMSC_WMQ_SEND_CHECK_COUNT*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

변환되지 않은 하나의 XMS 세션에서 비동기 put 오류를 검사하는 사이에 허용할 수 있는 송신 호출 수입니다.

기본적으로 이 특성은 0으로 설정됩니다.

### *XMSC_WMQ_SHARE_CONV_ALLOWED*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

클라이언트 연결이 채널 정의가 일치하는 경우 동일한 프로세스에서 동일한 큐 관리자로의 다른 최상위 레벨 XMS 연결과 소켓을 공유할 수 있는지 여부입니다. 이 특성은 애플리케이션 개발 또는 유지보수 또는 경영 목적을 위해 필요한 경우 별도의 소켓에서 연결의 완전한 격리를 허용하기 위해 제공됩니다. 이 특성을 설정하면 XMS에 기본 소켓이 공유됨을 나타냅니다. 이는 단일 소켓을 공유하는 연결 수를 표시하지는 않습니다. 소켓을 공유하는 연결의 수는 WebSphere MQ 클라이언트와 WebSphere MQ 서버 간에 협상되는 SHARECNV 값에 의해 판별됩니다.

애플리케이션은 다음과 같은 이름 지정된 상수로 이 특성을 설정할 수 있습니다.

- XMSC_WMQ_SHARE_CONV_ALLOWED_FALSE - 연결이 소켓을 공유하지 않습니다.
- XMSC_WMQ_SHARE_CONV_ALLOWED_TRUE - 연결이 소켓을 공유합니다.

기본적으로 이 특성은 XMSC_WMQ_SHARE_CONV_ALLOWED_ENABLED로 설정됩니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

### *XMSC_WMQ_SSL_CERT_STORES*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 인증서 폐기 목록(CRL)이 있는 서버의 위치입니다.

이 특성 값은 쉼표로 구분된 하나 이상의 URL 목록입니다. 각 URL의 형식은 다음과 같습니다.

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

이 형식은 호환 가능하지만 기본 MQJMS 형식에서 확장됩니다.

`serveraddress`를 빈 상태로 둘 수 있습니다. 이 경우, XMS는 값이 문자열 "localhost"라고 가정합니다.

목록의 예는 다음과 같습니다.

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 정보**
비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원
관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

### *XMSC_WMQ_SSL_CIPHER_SPEC*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

큐 관리자에 대한 보안 연결에 사용할 CipherSpec의 이름.

다음 표에 IBM WebSphere MQ SSL 및 TLS 지원과 함께 사용할 수 있는 암호 스펙이 나와 있습니다. 개인 인증서를 요청할 때 공용 및 개인 키 쌍의 키 크기를 지정합니다. SSL 데이터 교환 동안에 사용되는 키 크기는 인증서에 저장된 크기입니다(표에 명시된 것처럼 CipherSpec으로 판별되는 경우는 제외). 기본적으로 이 특성은 설정되어 있지 않습니다.

| CipherSpec 이름 | 사용되는 프로토콜 | 해시 알고리즘 | 암호화 알고리즘 | 암호화 비트 | FIPS[1] | 스위트 B 128 비트 | 스위트 B 192비트 |
|---|---|---|---|---|---|---|---|
| NULL_MD5 | SSL 3.0 | MD5 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| NULL_SHA | SSL 3.0 | SHA-1 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| RC4_MD5_EXPORT [2] | SSL 3.0 | MD5 | RC4 | 40 | 아니오 | 아니오 | 아니오 |
| RC4_MD5_US | SSL 3.0 | MD5 | RC4 | 128 | 아니오 | 아니오 | 아니오 |
| RC4_SHA_US | SSL 3.0 | SHA-1 | RC4 | 128 | 아니오 | 아니오 | 아니오 |
| RC2_MD5_EXPORT[2] | SSL 3.0 | MD5 | RC2 | 40 | 아니오 | 아니오 | 아니오 |

| CipherSpec 이름 | 사용되는 프로토콜 | 해시 알고리즘 | 암호화 알고리즘 | 암호화 비트 | FIPS[1] | 스위트 B 128 비트 | 스위트 B 192비트 |
|---|---|---|---|---|---|---|---|
| DES_SHA_EXPORT [2] | SSL 3.0 | SHA-1 | DES | 56 | 아니오 | 아니오 | 아니오 |
| RC4_56_SHA_EXPORT1024[3] | SSL 3.0 | SHA-1 | RC4 | 56 | 아니오 | 아니오 | 아니오 |
| DES_SHA_EXPORT1024 [3] | SSL 3.0 | SHA-1 | DES | 56 | 아니오 | 아니오 | 아니오 |
| TRIPLE_DES_SHA_US | SSL 3.0 | SHA-1 | 3DES | 168 | 아니오 | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_128_CBC_SHA | TLS 1.0 | SHA-1 | AES | 128 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_256_CBC_SHA[4] | TLS 1.0 | SHA-1 | AES | 256 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_DES_CBC_SHA | TLS 1.0 | SHA-1 | DES | 56 | 아니오[5] | 아니오 | 아니오 |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA[8] | TLS 1.0 | SHA-1 | 3DES | 168 | 예 | 아니오 | 아니오 |
| FIPS_WITH_DES_CBC_SHA | SSL 3.0 | SHA-1 | DES | 56 | 아니오[6] | 아니오 | 아니오 |
| FIPS_WITH_3DES_EDE_CBC_SHA | SSL 3.0 | SHA-1 | 3DES | 168 | 아니오[7] | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_128_GCM_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_256_GCM_SHA384 | TLS 1.2 | SHA-384 | AES | 256 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_128_CBC_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_AES_256_CBC_SHA256 | TLS 1.2 | SHA-256 | AES | 256 | 예 | 아니오 | 아니오 |
| ECDHE_ECDSA_RC4_128_SHA256 | TLS 1.2 | SHA-256 | RC4 | 128 | 아니오 | 아니오 | 아니오 |
| ECDHE_ECDSA_3DES_EDE_CBC_SHA256 | TLS 1.2 | SHA-256 | 3DES | 168 | 예 | 아니오 | 아니오 |
| ECDHE_RSA_RC4_128_SHA256 | TLS 1.2 | SHA-256 | RC4 | 128 | 아니오 | 아니오 | 아니오 |
| ECDHE_RSA_3DES_EDE_CBC_SHA256 | TLS 1.2 | SHA-256 | 3DES | 168 | 예 | 아니오 | 아니오 |
| ECDHE_ECDSA_AES_128_CBC_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 아니오 | 아니오 |
| ECDHE_ECDSA_AES_256_CBC_SHA384 | TLS 1.2 | SHA-384 | AES | 256 | 예 | 아니오 | 아니오 |
| ECDHE_RSA_AES_128_CBC_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 아니오 | 아니오 |
| ECDHE_RSA_AES_256_CBC_SHA384 | TLS 1.2 | SHA-384 | AES | 256 | 예 | 아니오 | 아니오 |
| ECDHE_ECDSA_AES_128_GCM_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 예 | 아니오 |
| ECDHE_ECDSA_AES_256_GCM_SHA384 | TLS 1.2 | SHA-384 | AES | 256 | 예 | 아니오 | 예 |

| CipherSpec 이름 | 사용되는 프로토콜 | 해시 알고리즘 | 암호화 알고리즘 | 암호화 비트 | FIPS[1] | 스위트 B 128 비트 | 스위트 B 192비트 |
|---|---|---|---|---|---|---|---|
| ECDHE_RSA_AES_128_GCM_SHA256 | TLS 1.2 | SHA-256 | AES | 128 | 예 | 아니오 | 아니오 |
| ECDHE_RSA_AES_256_GCM_SHA384 | TLS 1.2 | SHA-384 | AES | 256 | 예 | 아니오 | 아니오 |
| TLS_RSA_WITH_NULL_SHA256 | TLS 1.2 | SHA-256 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| ECDHE_RSA_NULL_SHA256 | TLS 1.2 | SHA-256 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| ECDHE_ECDSA_NULL_SHA256 | TLS 1.2 | SHA-256 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| TLS_RSA_WITH_NULL_NULL | TLS 1.2 | 없음 | 없음 | 0 | 아니오 | 아니오 | 아니오 |
| TLS_RSA_WITH_RC4_128_SHA256 | TLS 1.2 | SHA-256 | RC4 | 128 | 아니오 | 아니오 | 아니오 |

**참고사항:**

1. CipherSpec이 FIPS(Federal Information Processing Standard) 140-2를 준수하는지 여부를 지정합니다. FIPS에 대한 설명과 FIPS 140-2 준수 조작을 위해 WebSphere MQ를 구성하는 방법에 대한 정보를 보려면 온라인 IBM IBM WebSphere MQ 제품 문서에서 *FIPS(Federal Information Processing Standard)*를 참조하십시오.
2. 최대 데이터 교환 키 크기는 512비트입니다. SSL 데이터 교환 중에 교환된 인증서 중 한 개의 키 크기가 512비트를 초과할 경우, 데이터 교환 중에 사용할 수 있도록 임시 512비트 키가 생성됩니다.
3. 데이터 교환 키 크기는 1024비트입니다.
4. 적절한 무제한 정책 파일이 WebSphere MQ Explorer에서 사용되는 JRE에 적용되지 않는 경우 Explorer에서 큐 관리자로 보안 연결을 하는 데 이 CipherSpec을 사용할 수 없습니다.
5. 이 CipherSpec은 2007년 5월 19일 이전에 FIPS 140-2 인증되었습니다.
6. 이 CipherSpec은 2007년 5월 19일 이전에 FIPS 140-2 인증되었습니다. FIPS_WITH_DES_CBC_SHA라는 이름은 역사적인 의미의 이름으로, 이 CipherSpec이 이전에 FIPS를 준수했었다는 사실을 나타냅니다(현재는 더 이상 준수하지 않음). 이 CipherSpec은 더 이상 사용되지 않습니다.
7. FIPS_WITH_3DES_EDE_CBC_SHA라는 이름은 역사적인 의미의 이름으로, 이 CipherSpec이 이전에 FIPS를 준수했었다는 사실을 나타냅니다(현재는 더 이상 준수하지 않음). 이 CipherSpec은 더 이상 사용되지 않습니다.
8. WebSphere MQ가 FIPS 140-2 준수 조작을 위해 구성된 경우, 연결이 오류 AMQ9288과 함께 종료되기 전에 이 CipherSpec을 사용하여 최대 32GB의 데이터를 전송할 수 있습니다. 이 오류를 방지하려면 3중 DES(더 이상 사용되지 않음)를 사용하지 않거나, FIPS 140-2 구성에서 이 CipherSpec을 사용할 때 비밀 키 재설정을 사용으로 설정하십시오.

**관련 정보**

보안

메시지의 데이터 무결성

CipherSpec 지정

### *XMSC_WMQ_SSL_CIPHER_SUITE*

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

큐 관리자에 대한 SSL 또는 TLS 연결에 사용될 CipherSuite 의 이름입니다. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다.

이 특성의 표준 값은 다음과 같습니다.

- 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_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

이 값은 XMSC_WMQ_SSL_CIPHER_SPEC 대신 제공될 수 있습니다.

XMSC_WMQ_SSL_CIPHER_SPEC에 비어 있지 않은 값이 지정되는 경우 이 값이 XMSC_WMQ_SSL_CIPHER_SUITE에 대한 설정을 대체합니다. XMSC_WMQ_SSL_CIPHER_SPEC에 값이 없는 경우 XMSC_WMQ_SSL_CIPHER_SUITE의 값이 GSKit에 제공될 암호 스위트로 사용됩니다. 이 경우, 60 페이지의 『WebSphere MQ 큐 관리자에 대한 연결에 필요한 CipherSuite 및 CipherSpec 이름 맵핑』 에 설명된 대로 값이 해당 CipherSpec 값에 맵핑됩니다.

XMSC_WMQ_SSL_CIPHER_SPEC 및 XMSC_WMQ_SSL_CIPHER_SUITE가 모두 비어 있는 경우 pChDef->SSLCipherSpec 필드가 공백으로 채워집니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

## *XMSC_WMQ_SSL_CRYPTO_HW*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항입니다.

이 특성의 표준 값은 다음과 같습니다.

- GSK_ACCELERATOR_RAINBOW_CS_OFF
- GSK_ACCELERATOR_RAINBOW_CS_ON
- GSK_ACCELERATOR_NCIPHER_NF_OFF
- GSK_ACCELERATOR_NCIPHER_NF_ON

PKCS11 암호화 하드웨어에 대한 특수한 형식이 있습니다(DriverPath, TokenLabel 및 TokenPassword가 사용자 지정 문자열임).

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS가 문자열 컨텐츠를 해석하거나 변경하지 않습니다. 최대 256개의 1바이트 문자로 제공된 값을 MQSCO.CryptoHardware 필드에 복사합니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

## *XMSC_WMQ_SSL_FIPS_REQUIRED*

**데이터 유형:**
부울

**특성:**
ConnectionFactory

이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 또는 사용할 수 없는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 FIPS 알고리즘만 클라이언트-서버 연결에 사용됩니다.

이 특성 값은 다음과 같으며 MQSCO.FipsRequired의 두 표준 값으로 변환됩니다.

표 *35. MQSCO.FlipsRequired* 특성 값 표

| 가치 | 설명 | 해당하는 MQSCO.FipsRequired 값 |
|---|---|---|
| false | 모든 CipherSpec을 사용할 수 있습니다. | MQSSL_FIPS_NO(기본값) |
| true | 이 클라이언트 연결에 적용되는 CipherSpec에서는 FIPS 인증 암호화 알고리즘만 사용할 수 있습니다. | MQSSL_FIPS_YES |

XMS는 MQCONNX 호출 전에 해당 값을 MQSCO.FipsRequired에 복사합니다.

매개변수 MQSCO.FipsRequired 는 WebSphere MQ 버전 6에서만 사용할 수 있습니다. WebSphere MQ 버전 5.3인 경우 이 특성이 설정되면 XMS에서 큐 관리자에 대한 연결을 시도하지 않으며 대신 적절하게 예외 처리합니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

## *XMSC_WMQ_SSL_KEY_REPOSITORY*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

키 및 인증서가 저장되는 키 데이터베이스 파일의 위치입니다.

XMS는 최대 256개의 1바이트 문자로 된 문자열을 MQSCO.KeyRepository 필드에 복사합니다. WebSphere MQ는 전체 경로를 포함하여 파일 이름으로 이 문자열을 해석합니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

## *XMSC_WMQ_SSL_KEY_RESETCOUNT*

**데이터 유형:**

System.Int32

**특성:**

ConnectionFactory

KeyResetCount는 비밀 키가 재협상되기 전에 SSL 통신에서 송신되고 수신된 암호화되지 않은 총 바이트 수를 나타냅니다. 바이트 수에는 MCA에서 전송된 제어 정보가 포함됩니다.

XMS는 MQCONNX 호출 전 사용자가 이 특성에 대해 제공한 값을 MQSCO.KeyResetCount로 복사합니다.

MQSCO.KeyRestCount 매개변수는 WebSphere MQ 버전 6에서만 사용할 수 있습니다. WebSphere MQ 버전 5.3인 경우 이 특성이 설정되면 XMS에서 큐 관리자에 대한 연결을 시도하지 않으며 대신 적절하게 예외 처리합니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

이 특성의 기본값은 0입니다. 즉, 시크릿 키가 재조정되지 않습니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

## *XMSC_WMQ_SSL_PEER_NAME*

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 피어 이름입니다.

이 특성은 표준 값 목록이 없습니다. 대신 SSLPEER에 대한 규칙에 따라 이 문자열을 빌드해야 합니다.

다음은 피어 이름의 예입니다.

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS는 문자열을 올바른 1바이트 코드 페이지로 복사하고 MQCONNX를 호출하기 전에 올바른 값을 MQCD.SSLPeerNamePtr 및 MQCD.SSLPeerNameLength에 배치합니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

.NET 전용: IBM MQ 8.0에서 IBM MQ(WMQ_CM_CLIENT)에 대한 관리 연결 및 IBM MQ(WMQ_CM_CLIENT_UNMANAGED)에 대한 비관리 연결 모두 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 정보**

비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

관리 .NET 클라이언트에 대한 SSL 및 TLS 지원

SSLPEERNAME

### *XMSC_WMQ_SYNCPOINT_ALL_GETS*

**데이터 유형:**
System.Boolean

**특성:**
ConnectionFactory

동기점 제어에 있는 큐에서 모든 메시지를 검색해야 하는지 여부를 결정합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| false | 상황이 적절한 경우 XMS 클라이언트는 동기점 제어 외부에 있는 큐에서 메시지를 검색할 수 있습니다. |
| true | XMS 클라이언트가 동기점 제어 내의 큐에서 모든 메시지를 검색해야 합니다. |

기본값은 false입니다.

### *XMSC_WMQ_TARGET_CLIENT*

**데이터 유형:**
System.Int32

**특성:**
목적지

**URI에서 사용되는 이름:**
targetClient

목적지에 송신되는 메시지에 MQRFH2 헤더를 포함할지 여부를 지정합니다.

애플리케이션이 MQRFH2 헤더를 포함하는 메시지를 전송하는 경우 수신하는 애플리케이션이 헤더를 처리할 수 있어야 합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
|---|---|
| XMSC_WMQ_TARGET_DEST_JMS | 목적지에 전송되는 메시지가 MQRFH2 헤더를 포함합니다. 애플리케이션이 다른 XMS 애플리케이션, WebSphere JMS 애플리케이션 또는 MQRFH2 헤더를 처리하도록 설계된 고유 WebSphere MQ 애플리케이션으로 메시지를 전송하는 경우 이 값을 지정하십시오. |
| XMSC_WMQ_TARGET_DEST_MQ | 목적지에 전송되는 메시지가 MQRFH2 헤더를 포함하지 않습니다. 애플리케이션이 MQRFH2 헤더를 처리하도록 설계되지 않은 고유 WebSphere MQ 애플리케이션으로 메시지를 전송하는 경우 이 값을 지정하십시오. |

기본값은 XMSC_WMQ_TARGET_DEST_JMS입니다.

### *XMSC_WMQ_TEMP_Q_PREFIX*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 XMS 임시 큐를 작성할 때 작성되는 WebSphere MQ 동적 큐의 이름을 구성하는 데 사용되는 접두부입니다.

The rules for forming the prefix are the same as the rules for forming the contents of the **DynamicQName** field in an object descriptor, but the last non-blank character must be an asterisk(*). If the property is not

set, the value used is CSQ.* on z/OS and AMQ.* on the other platforms. 기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 포인트-투-포인트 도메인에서만 관련됩니다.

### *XMSC_WMQ_TEMP_TOPIC_PREFIX*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory, Destination

임시 주제를 작성할 때 XMS 는 "TEMP/TEMPTOPICPREFIX/unique_id" 양식의 주제 문자열을 생성하거나 이 특성에 기본값이 포함된 경우 "TEMP/unique_id" 문자열이 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다.

이 특성의 올바른 값은 IBM WebSphere MQ 토픽 문자열에 유효한 문자만으로 구성된 널이 아닌 문자열입니다.

기본적으로 이 특성은 ""(빈 문자열)로 설정됩니다.

**참고:** 이 특성은 발행/구독 도메인에만 관련됩니다.

### *XMSC_WMQ_TEMPORARY_MODEL*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 XMS 임시 큐를 작성할 때 동적 큐가 작성되는 WebSphere MQ 모델 큐의 이름입니다.

이 특성의 기본값은 SYSTEM.DEFAULT.MODEL.QUEUE입니다.

이 특성은 포인트-투-포인트 도메인에서만 관련됩니다.

### *XMSC_WMQ_WILDCARD_FORMAT*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory, Destination

이 특성은 사용되는 와일드카드 구문 버전을 판별합니다.

IBM WebSphere MQ '*' 및 '?' 를 사용하여 발행/구독을 사용하는 경우 와일드카드로 처리합니다. 반면 '#' 및 '+'는 IBM Integration Bus에서 발행/구독을 사용할 때 와일드카드로 처리됩니다. 이 특성은 XMSC_WMQ_BROKER_VERSION 특성을 대체합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC_WMQ_WILDCARD_TOPIC_ONLY**
토픽 레벨 와일드카드만을 인식합니다. ' # ' '+' 는 와일드카드로 처리됩니다. 이 값은 XMSC_WMQ_BROKER_V2와 동일합니다.

**XMSC_WMQ_WILDCARD_CHAR_ONLY**
문자 와일드 카드만을 인식합니다. ' * ' 그리고 '? ' 와일드카드로 처리합니다. 이 값은 XMSC_WMQ_BROKER_V1과 동일합니다.

기본적으로 이 특성은 XMSC_WMQ_WILDCARD_TOPIC_ONLY로 설정됩니다.

### *XMSC_WPM_BUS_NAME*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**
busName

연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다.

토픽인 목적지의 경우 이 특성은 연관된 토픽 공간이 존재하는 서비스 통합 버스의 이름입니다. 이 토픽 공간은 XMSC_WPM_TOPIC_SPACE 특성에 의해 지정됩니다.

특성에 목적지가 설정되지 않으면 큐 또는 연관된 토픽 공간이 애플리케이션이 연결하는 서비스 통합 버스에 존재한다고 가정합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_CONNECTION_PROTOCOL*

**데이터 유형:**
System.Int32

**특성:**
연결

메시징 엔진에 연결하는 데 사용되는 통신 프로토콜입니다. 이 특성은 읽기 전용입니다.

이 특성의 가능한 값은 다음과 같습니다.

| 가치 | 의미 |
|---|---|
| XMSC_WPM_CP_HTTP | 연결이 TCP/IP에서 HTTP를 사용합니다. |
| XMSC_WPM_CP_TCP | 연결이 TCP/IP를 사용합니다. |

### *XMSC_WPM_CONNECTION_PROXIMITY*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

연결에 대한 연결 근접성 설정입니다. 이 특성은 애플리케이션이 연결되는 메시징 엔진이 부트스트랩 서버에 얼마나 가까워야 하는지를 결정합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 연결 근접성 설정 |
|---|---|
| XMSC_WPM_CONNECTION_PROXIMITY_BUS | 버스 |
| XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER | 군집 |
| XMSC_WPM_CONNECTION_PROXIMITY_HOST | 호스트 |
| XMSC_WPM_CONNECTION_PROXIMITY_SERVER | SERVER |

기본값은 XMSC_WPM_CONNECTION_PROXIMITY_BUS입니다.

### *XMSC_WPM_DUR_SUB_HOME*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

**URI에서 사용되는 이름:**
durableSubscriptionHome

목적지나 연결에 대한 모든 지속 가능한 구독을 관리하는 메시징 엔진의 이름입니다. 지속 가능 구독자에 전달될 메시지는 동일한 메시징 엔진의 발행 위치에 저장됩니다.

연결에 대한 지속 가능 구독 홈을 지정해야 애플리케이션에서 연결을 사용하는 지속 가능 구독자를 작성할 수 있습니다. 목적지에 지정한 값은 연결에 지정한 값을 대체합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WPM_HOST_NAME*

**데이터 유형:**
문자열

**특성:**
연결

애플리케이션이 연결되는 메시징 엔진이 포함된 시스템의 호스트 이름 또는 IP 주소입니다. 이 특성은 읽기 전용입니다.

### *XMSC_WPM_LOCAL_ADDRESS*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

서비스 통합 버스에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.

이 특성의 값은 다음 형식의 문자열입니다.

[*host_name*][(*low_port*)[,*high_port*])]

변수의 의미는 다음과 같습니다.

***host_name***
연결에 사용하는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 정보는 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 필요하며 사용자가 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이고 사용해야 할 인터페이스를 지정하지 않으면 인터페이스가 임의로 선택됩니다.

***low_port***
연결에 사용되는 로컬 포트의 번호입니다.

*high_port*도 함께 지정하면 *low_port*가 포트 번호 범위 중에서 가장 낮은 번호로 해석됩니다.

***high_port***
포트 번호 범위에서 가장 높은 포트 번호입니다. 지정된 범위의 포트 중 하나를 연결에 사용해야 합니다.

다음은 특성의 올바른 값에 대한 예입니다.

JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_ME_NAME*

**데이터 유형:**
문자열

**특성:**
연결

애플리케이션이 연결되는 메시징 엔진의 이름입니다. 이 특성은 읽기 전용입니다.

### *XMSC_WPM_NON_PERSISTENT_MAP*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

연결을 사용하여 송신되는 비지속적 메시지의 신뢰도 레벨입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 신뢰도 레벨 |
| --- | --- |
| XMSC_WPM_MAPPING_AS_DESTINATION | 서비스 통합 버스의 큐 또는 토픽 공간에 지정된 기본 신뢰도 레벨로 판별됨 |
| XMSC_WPM_MAPPING_BEST_EFFORT_NON_PERSISTENT | 최상의 비지속적 상태 |
| XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT | 명확한 비지속적 상태 |
| XMSC_WPM_MAPPING_RELIABLE_NON_PERSISTENT | 신뢰 가능한 비지속적 상태 |
| XMSC_WPM_MAPPING_RELIABLE_PERSISTENT | 신뢰 가능한 지속적 상태 |
| XMSC_WPM_MAPPING_ASSURED_PERSISTENT | 확실한 지속적 상태 |

기본값은 XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT입니다.

### *XMSC_WPM_PERSISTENT_MAP*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

연결을 사용하여 송신되는 지속 메시지의 신뢰도 레벨입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 신뢰도 레벨 |
| --- | --- |
| XMSC_WPM_MAPPING_AS_DESTINATION | 서비스 통합 버스의 큐 또는 토픽 공간에 지정된 기본 신뢰도 레벨로 판별됨 |
| XMSC_WPM_MAPPING_BEST_EFFORT_NON_PERSISTENT | 최상의 비지속적 상태 |

| 올바른 값 | 신뢰도 레벨 |
| --- | --- |
| XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT | 명확한 비지속적 상태 |
| XMSC_WPM_MAPPING_RELIABLE_NON_PERSISTENT | 신뢰 가능한 비지속적 상태 |
| XMSC_WPM_MAPPING_RELIABLE_PERSISTENT | 신뢰 가능한 지속적 상태 |
| XMSC_WPM_MAPPING_ASSURED_PERSISTENT | 확실한 지속적 상태 |

기본값은 XMSC_WPM_MAPPING_RELIABLE_PERSISTENT입니다.

### *XMSC_WPM_PORT*

**데이터 유형:**
System.Int32

**특성:**
연결

애플리케이션이 연결되는 메시징 엔진이 대기하는 포트의 번호입니다. 이 특성은 읽기 전용입니다.

### *XMSC_WPM_PROVIDER_ENDPOINTS*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

부트스트랩 서버에 사용되는 하나 이상의 엔드포인트 주소의 순서입니다. 엔드포인트 주소는 쉼표로 구분됩니다.

부트스트랩 서버는 애플리케이션이 연결되는 메시징 엔진을 선택하는 애플리케이션 서버입니다. 부트스트랩 서버의 엔드포인트 주소 형식은 다음과 같습니다.

*host_name*:*port_number*:*chain_name*

엔드포인트 주소 구성요소의 의미는 다음과 같습니다.

***host_name***
부트스트랩 서버가 상주하는 시스템의 호스트 이름 또는 IP 주소입니다. 호스트 이름 또는 IP 주소가 지정되지 않은 경우 기본값은 `localhost`입니다.

***port_number***
부트스트랩 서버가 수신 요청을 청취하는 포트 번호입니다. 포트 번호가 지정되지 않은 경우 기본값은 7276입니다.

***chain_name***
부트스트랩 서버에서 사용하는 부트스트랩 전송 체인의 이름입니다. 올바른 값은 다음과 같습니다.

| 올바른 값 | 부트스트랩 전송 체인의 이름 |
| --- | --- |
| XMSC_WPM_BOOTSTRAP_HTTP | BootstrapTunneledMessaging |
| XMSC_WPM_BOOTSTRAP_HTTPS | BootstrapTunneledSecureMessaging |
| XMSC_WPM_BOOTSTRAP_SSL | BootstrapSecureMessaging |
| XMSC_WPM_BOOTSTRAP_TCP | BootstrapBasicMessaging |

이름이 지정되지 않은 경우 기본값은 XMSC_WPM_BOOTSTRAP_TCP입니다.

엔드포인트 주소가 지정되지 않은 경우 기본값은 localhost:7276:BootstrapBasicMessaging입니다.

### *XMSC_WPM_SSL_CIPHER_SUITE*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

WebSphere 서비스 통합 버스 메시징 엔진에 대한 SSL 또는 TLS 연결에 사용할 CipherSuite의 이름. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다.

표 *36. WebSphere* 서비스 통합 버스 메시징 엔진에 연결하는 데 필요한 *CipherSuite* 옵션

| **암호화 스위트** | **사용되는 프로토콜** |
|---|---|
| SSL_RSA_WITH_NULL_MD5 | SSLv3 |
| SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5 | SSLv3 |
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | SSLv3 |
| SSL_RSA_WITH_RC4_128_MD5 | SSLv3 |
| SSL_RSA_WITH_NULL_SHA | SSLv3 |
| SSL_RSA_EXPORT1024_WITH_RC4_56_SHA | SSLv3 |
| SSL_RSA_WITH_RC4_128_SHA | SSLv3 |
| SSL_RSA_WITH_DES_CBC_SHA | SSLv3 |
| SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA | SSLv3 |
| SSL_RSA_FIPS_WITH_DES_CBC_SHA | SSLv3 |
| SSL_RSA_WITH_3DES_EDE_CBC_SHA | SSLv3 |
| SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA | SSLv3 |
| TLS_RSA_WITH_DES_CBC_SHA | TLSv1 |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA(더 이상 사용되지 않음) | TLSv1 |
| TLS_RSA_WITH_AES_128_CBC_SHA | TLSv1 |
| TLS_RSA_WITH_AES_256_CBC_SHA | TLSv1 |

**참고:** TLS_RSA_WITH_AES_128_CBC_SHA 및 TLS_RSA_WITH_AES_256_CBC_SHA CipherSuites는 Windows 또는 Solaris에서만 지원됩니다 (이는 GSKit에 의해 지정됨).

이 특성에는 기본값이 없습니다. SSL 또는 TLS를 사용하려면 이 특성에 대한 값을 지정해야 하고, 그렇지 않을 경우 애플리케이션이 서버에 연결할 수 없습니다.

### *XMSC_WPM_SSL_FIPS_REQUIRED*

**데이터 유형:**
부울

**특성:**
ConnectionFactory

이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 또는 사용할 수 없는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 클라이언트-서버 연결에 FIPS 알고리즘만 사용됩니다.이 특성의 값을 TRUE로 설정하면 애플리케이션이 비FIPS 준수 암호 스위트를 사용하지 않도록 방지합니다.

기본적으로 이 특성은 FALSE(즉, FIPS 모드 사용 안함)로 설정되어 있습니다.

### *XMSC_WPM_SSL_KEY_REPOSITORY*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

보안 연결에 사용되는 공개 또는 개인 키가 포함되어 있는 키 링 파일의 파일 경로입니다.

키 링 파일 특성을 XMSC_WPM_SSL_MS_CERTIFICATE_STORE의 특수 값으로 설정하면 Microsoft Windows 키 데이터베이스를 사용하도록 지정하는 것입니다. Microsoft Windows 키 데이터베이스(**제어판 > 인터넷 옵션 > 컨텐츠 > 인증서**에 위치)를 사용하면 별도의 키 파일 데이터베이스에 대한 필요가 없어집니다. Windows x64 및 기타 플랫폼에서는 이 상수를 사용할 수 없습니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_SSL_KEYRING_LABEL*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

서버 인증 시 사용되는 인증서입니다. 값이 지정되지 않은 경우 기본 인증서가 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_SSL_KEYRING_PW*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

키 링 파일의 비밀번호입니다.

이 특성을 XMSC_WPM_SSL_KEYRING_STASH_FILE 대신 사용하여 키 링 파일의 비밀번호를 구성할 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_SSL_KEYRING_STASH_FILE*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

키 저장소 파일의 비밀번호가 있는 2진 파일의 이름입니다.

이 특성을 XMSC_WPM_SSL_KEYRING_PW 대신 사용하여 키 링 파일의 비밀번호를 구성할 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_TARGET_GROUP*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

메시징 엔진의 대상 그룹 이름입니다. 대상 그룹의 네이처는 XMSC_WPM_TARGET_TYPE 특성으로 판별됩니다.

서비스 통합 버스에서 메시징 엔진의 하위 그룹으로 메시징 엔진 검색을 제한하려면 이 특성을 설정하십시오. 애플리케이션이 서비스 통합 버스에서 메시징 엔진에 연결할 수 있도록 하려면 이 특성을 설정하지 마십시오.

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_TARGET_SIGNIFICANCE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

메시징 엔진의 대상 그룹에 대한 중요성입니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WPM_TARGET_SIGNIFICANCE_ PREFERRED | 사용 가능한 경우 대상 그룹에서 메시징 엔진이 선택됩니다. 그렇지 않으면 동일한 서비스 통합 버스에서 대상 그룹 외부의 메시징 엔진이 선택됩니다. |
| XMSC_WPM_TARGET_SIGNIFICANCE_ REQUIRED | 선택된 메시징 엔진은 대상 그룹에 있어야 합니다. 대상 그룹의 메시징 엔진이 사용 가능하지 않으면 연결 프로세스가 실패합니다. |

이 특성의 기본값은 XMSC_WPM_TARGET_SIGNIFICANCE_PREFERRED입니다.

### *XMSC_WPM_TARGET_TRANSPORT_CHAIN*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 메시징 엔진에 연결하는 데 사용해야 하는 인바운드 전송 체인의 이름입니다.

이 특성의 값은 메시징 엔진을 호스트하는 애플리케이션 서버에서 사용 가능한 인바운드 전송 체인의 이름일 수 있습니다. 다음 이름 지정된 상수가 사전정의된 인바운드 전송 체인 중 하나에 제공됩니다.

| 이름 지정된 상수 | 전송 체인의 이름 |
| --- | --- |
| XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC | InboundBasicMessaging |

이 특성의 기본값은 XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC입니다.

### *XMSC_WPM_TARGET_TYPE*

**데이터 유형:**
System.Int32

**특성:**
ConnectionFactory

메시징 엔진의 대상 그룹 유형입니다. 이 특성은 XMSC_WPM_TARGET_GROUP 특성으로 식별되는 대상 그룹의 속성을 판별합니다.

특성의 올바른 값은 다음과 같습니다.

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WPM_TARGET_TYPE_BUSMEMBER | 대상 그룹의 이름은 버스 멤버의 이름입니다. 대상 그룹은 버스 멤버의 모든 메시징 엔진입니다. |
| XMSC_WPM_TARGET_TYPE_CUSTOM | 대상 그룹의 이름은 메시징 엔진의 사용자 정의 그룹 이름입니다. 대상 그룹은 사용자 정의 그룹에 등록된 모든 메시징 엔진입니다. |

| 올바른 값 | 의미 |
| --- | --- |
| XMSC_WPM_TARGET_TYPE_ME | 대상 그룹의 이름은 메시징 엔진의 이름입니다. 대상 그룹은 지정된 메시징 엔진입니다. |

기본적으로 특성이 설정되어 있지 않습니다.

### *XMSC_WPM_TEMP_Q_PREFIX*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 XMS 임시 큐를 작성할 때 서비스 통합 버스에 작성되는 임시 큐의 이름을 구성하는 데 사용되는 접두부입니다. 접두부에는 최대 12자가 포함될 수 있습니다.

임시 큐의 이름은 "_Q" 로 시작하고 그 뒤에 접두부가 붙습니다. 이름의 나머지 부분은 시스템 생성 문자로 구성됩니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 임시 큐의 이름에 접두부가 없음을 의미합니다.

이 특성은 포인트-투-포인트 도메인에서만 관련됩니다.

### *XMSC_WPM_TEMP_TOPIC_PREFIX*

**데이터 유형:**
문자열

**특성:**
ConnectionFactory

애플리케이션이 작성한 임시 토픽의 이름을 구성하는 데 사용되는 접두부입니다. 접두부에는 최대 12자가 포함될 수 있습니다.

임시 토픽의 이름은 "_T" 로 시작하고 그 뒤에 접두부가 붙습니다. 이름의 나머지 부분은 시스템 생성 문자로 구성됩니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 임시 토픽의 이름에 접두부가 없음을 의미합니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

### *XMSC_WPM_TOPIC_SPACE*

**데이터 유형:**
문자열

**특성:**
목적지

**URI에서 사용되는 이름:**
topicSpace

토픽을 포함하는 토픽 영역의 이름입니다. 토픽인 목적지만 이 특성을 보유할 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 기본 토픽 공간이 가정됨을 의미합니다.

이 특성은 발행/구독 도메인에서만 관련됩니다.

# 주의사항

이 정보는 미국에서 제공되는 제품 및 서비스용으로 작성된 것입니다.

IBM은 다른 국가에서 이 책에 기술된 제품, 서비스 또는 기능을 제공하지 않을 수도 있습니다. 현재 사용할 수 있는 제품 및 서비스에 대한 정보는 한국 IBM 담당자에게 문의하십시오. 이 책에서 IBM 제품, 프로그램 또는 서비스를 언급했다고 해서 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상으로 동등한 제품, 프로그램 또는 서비스를 대신 사용할 수도 있습니다. 그러나 비IBM 제품, 프로그램 또는 서비스의 운영에 대한 평가 및 검증은 사용자의 책임입니다.

IBM은 이 책에서 다루고 있는 특정 내용에 대해 특허를 보유하고 있거나 현재 특허 출원 중일 수 있습니다. 이 책을 제공한다고 해서 특허에 대한 라이센스까지 부여하는 것은 아닙니다. 라이센스에 대한 의문사항은 다음으로 문의하십시오.

150-945
서울특별시 영등포구
국제금융로 10, 3IFC
한국 아이.비.엠 주식회사
U.S.A.

2바이트(DBCS) 정보에 관한 라이센스 문의는 한국 IBM에 문의하거나 다음 주소로 서면 문의하시기 바랍니다.

지적 재산권 라이센스 부여
2-31 Roppongi 3-chome, Minato-Ku
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

**다음 단락은 현지법과 상충하는 영국이나 기타 국가에서는 적용되지 않습니다.** IBM은 타인의 권리 비침해, 상품성 및 특정 목적에의 적합성에 대한 묵시적 보증을 포함하여(단, 이에 한하지 않음) 명시적 또는 묵시적인 일체의 보증 없이 이 책을 "현상태대로" 제공합니다. 일부 국가에서는 특정 거래에서 명시적 또는 묵시적 보증의 면책사항을 허용하지 않으므로, 이 사항이 적용되지 않을 수도 있습니다.

이 정보에는 기술적으로 부정확한 내용이나 인쇄상의 오류가 있을 수 있습니다. 이 정보는 주기적으로 변경되며, 변경된 사항은 최신판에 통합됩니다. IBM은 이 책에서 설명한 제품 및/또는 프로그램을 사전 통지 없이 언제든지 개선 및/또는 변경할 수 있습니다.

이 정보에서 언급되는 비IBM의 웹 사이트는 단지 편의상 제공된 것으로, 어떤 방식으로든 이들 웹 사이트를 옹호하고자 하는 것은 아닙니다. 해당 웹 사이트의 자료는 본 IBM 제품 자료의 일부가 아니므로 해당 웹 사이트 사용으로 인한 위험은 사용자 본인이 감수해야 합니다.

IBM은 귀하의 권리를 침해하지 않는 범위 내에서 적절하다고 생각하는 방식으로 귀하가 제공한 정보를 사용하거나 배포할 수 있습니다.

(i) 독립적으로 작성된 프로그램과 기타 프로그램(본 프로그램 포함) 간의 정보 교환 및 (ii) 교환된 정보의 상호 이용을 목적으로 본 프로그램에 관한 정보를 얻고자 하는 라이센스 사용자는 다음 주소로 문의하십시오.

서울특별시 영등포구
서울특별시 강남구 도곡동 467-12,
군인공제회관빌딩
한국 아이.비.엠 주식회사
U.S.A.

이러한 정보는 해당 조건(예를 들면, 사용료 지불 등)하에서 사용될 수 있습니다.

이 정보에 기술된 라이센스가 부여된 프로그램 및 프로그램에 대해 사용 가능한 모든 라이센스가 부여된 자료는 IBM이 IBM 기본 계약, IBM 프로그램 라이센스 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

본 문서에 포함된 모든 성능 데이터는 제한된 환경에서 산출된 것입니다. 따라서 다른 운영 환경에서 얻어진 결과는 상당히 다를 수 있습니다. 일부 성능은 개발 단계의 시스템에서 측정되었을 수 있으므로 이러한 측정치가 일반적으로 사용되고 있는 시스템에서도 동일하게 나타날 것이라고는 보증할 수 없습니다. 또한 일부 성능은 추정을

통해 추측되었을 수도 있으므로 실제 결과는 다를 수 있습니다. 이 책의 사용자는 해당 데이터를 본인의 특정 환경에서 검증해야 합니다.

비IBM 제품에 관한 정보는 해당 제품의 공급업체, 공개 자료 또는 기타 범용 소스로부터 얻은 것입니다. IBM에서는 이러한 제품들을 테스트하지 않았으므로, 비IBM 제품과 관련된 성능의 정확성, 호환성 또는 기타 청구에 대해서는 확신할 수 없습니다. 비IBM 제품의 성능에 대한 의문사항은 해당 제품의 공급업체에 문의하십시오.

IBM이 제시하는 방향 또는 의도에 관한 모든 언급은 특별한 통지 없이 변경될 수 있습니다.

이 정보에는 일상의 비즈니스 운영에서 사용되는 자료 및 보고서에 대한 예제가 들어 있습니다. 이들 예제에는 개념을 가능한 완벽하게 설명하기 위하여 개인, 회사, 상표 및 제품의 이름이 사용될 수 있습니다. 이들 이름은 모두 가공의 것이며 실제 기업의 이름 및 주소와 유사하더라도 이는 전적으로 우연입니다.

저작권 라이센스:

이 정보에는 여러 운영 플랫폼에서의 프로그래밍 기법을 보여주는 원어로 된 샘플 응용프로그램이 들어 있습니다. 귀하는 이러한 샘플 프로그램의 작성 기준이 된 운영 플랫폼의 응용프로그램 프로그래밍 인터페이스(API)에 부합하는 응용프로그램을 개발, 사용, 판매 또는 배포할 목적으로 IBM에 추가 비용을 지불하지 않고 이들 샘플 프로그램을 어떠한 형태로든 복사, 수정 및 배포할 수 있습니다. 이러한 샘플 프로그램은 모든 조건하에서 완전히 테스트된 것은 아닙니다. 따라서 IBM은 이들 샘플 프로그램의 신뢰성, 서비스 가능성 또는 기능을 보증하거나 진술하지 않습니다.

이 정보를 소프트카피로 확인하는 경우에는 사진과 컬러 삽화가 제대로 나타나지 않을 수도 있습니다.

## 프로그래밍 인터페이스 정보

프로그래밍 인터페이스 정보는 본 프로그램과 함께 사용하기 위한 응용프로그램 소프트웨어 작성을 돕기 위해 제공됩니다.

이 책에는 고객이 프로그램을 작성하여 WebSphere MQ서비스를 얻을 수 있도록 하는 계획된 프로그래밍 인터페이스에 대한 정보가 포함되어 있습니다.

그러나 본 정보에는 진단, 수정 및 성능 조정 정보도 포함되어 있습니다. 진단, 수정 및 성능 조정 정보는 응용프로그램 소프트웨어의 디버그를 돕기 위해 제공된 것입니다.

**중요사항:** 이 진단, 수정 및 튜닝 정보는 변경될 수 있으므로 프로그래밍 인터페이스로 사용하지 마십시오.

## 상표

IBM, IBM 로고, ibm.com®는 전세계 여러 국가에 등록된 IBM Corporation의 상표입니다. 현재 IBM 상표 목록은 웹 "저작권 및 상표 정보"(www.ibm.com/legal/copytrade.shtml) 에 있습니다. 기타 제품 및 서비스 이름은 IBM 또는 타사의 상표입니다.

Microsoft 및 Windows는 미국 또는 기타 국가에서 사용되는 Microsoft Corporation의 상표입니다.

UNIX는 미국 또는 기타 국가에서 사용되는 The Open Group의 등록상표입니다.

Linux®는 미국 또는 기타 국가에서 사용되는 Linus Torvalds의 등록상표입니다.

이 제품에는 Eclipse 프로젝트 (http://www.eclipse.org/) 에서 개발한 소프트웨어가 포함되어 있습니다.

Java 및 모든 Java 기반 상표와 로고는 Oracle 및/또는 그 계열사의 상표 또는 등록상표입니다.

부품 번호:

(1P) P/N: