7.5

IBM WebSphere MQ İçin Uygulama Geliştirilmesi

Not

Bu bilgileri ve desteklediği ürünü kullanmadan önce, “Özel notlar” sayfa 1081 bölümündeki bilgileri okuyun.

Bu basım, yeni basımlarında tersi belirtilmediği sürece, IBM® WebSphere MQ ' ın 7. yayın düzeyi 5 'i ve sonraki tüm yayın ve değişiklik düzeyleri için geçerlidir.

İçindekiler

Uygulamaların geliştirilmesi

IBM WebSphere MQ , iş süreçlerinizi desteklemek için gereksinim duyardığınız iletileri göndermek ve almak için uygulamalar geliştirebileceğiniz çeşitli yollar sağlar. Kuyruk yöneticilerinizi ve ilgili kaynaklarınızı yönetmek için de uygulamalar geliştirebilirsiniz.

IBM WebSphere MQiçin uygulama geliştirmeden önce, IBM WebSphere MQ Teknik genel bakışIBM WebSphere MQ Teknik genel bakışbaşlıklı konulardaki kavramlara aşina olduğunuzu doğrulayın.

IBM WebSphere MQ için farklı programlama dillerinde bir dizi uygulama geliştirebilirsiniz. Desteklenen programlama dillerine ve özelliklerine ilişkin bilgi için bkz. “Kullanılacak programlama diline karar verme” sayfa 75.

Farklı altyapılarda IBM WebSphere MQ için yazabileceğiniz uygulama tipleri için aşağıdaki kısımlara bakın.

Types of application you can write for IBM WebSphere MQ

Bu bilgiler, IBM WebSphere MQ' ta yazılabilecek uygulamaların tipleriyle ilgilidir.

IBM WebSphere MQ ürünleri, kuyruk yöneticileridir ve uygulama etkinleştiricileridir. Bu programlar,programsMessage Queue Interface (MQI) ' ı (Message Queue Interface; IBM Message Queue Interface; İleti Kuyruğu Arabirimi) destekler. Bu arabirim, bir kuyruğa ileti yerleştirip kuyru

z/OS dışı altyapılar için IBM WebSphere MQ ile aşağıdaki uygulamaları yazabilirsiniz:

- Aynı işletim sistemleri altında çalışan diğer uygulamalara ileti gönderme. Uygulamalar aynı ya da başka bir sistemde olabilir.
- Diğer IBM WebSphere MQ platformlarında çalışan uygulamalara ileti gönderin.
- Use message queuing from within CICS for TXSeries for AIX, TXSeries for HP-UX, TXSeries for Solaris, and TXSeries for Windows systems applications.
- Use message queuing from within Encina for AIX, HP-UX, Solaris, and Pencereler systems.
- Use message queuing from within Tuxedo for AIX, AT&T, HP-UX, Solaris, and Pencereler systems.
- Use IBM WebSphere MQ as a transaction manager, coordinating updates made by external resource managers within IBM WebSphere MQ units of work. Aşağıdaki dış kaynak yöneticileri desteklenir ve X/XX_ENCODE_CASE_ONE open XA arabirimiyle uyumludur.
  - DB2
  - Informix
  - Oracle
  - Sybase
- Birden çok iletiyi, kesinleştirilebilecek ya da yedeklenebilecek tek bir iş birimi olarak işler.
- Tam IBM WebSphere MQ ortamından çalıştır ya da aşağıdaki altyapılarda bir IBM WebSphere MQ MQI istemci ortamından çalıştırın:
  - UNIX and Linux® sistemleri
  - Pencereler

İlgili kavramlar

Güvenlik

Uygulama geliştirme kavramları

IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

IBM WebSphere MQ uygulamalarınızı tasarlamaya ve yazmaya başlamadan önce, temel IBM WebSphere MQ kavramlarını tanıyın ve Teknik genel bakışbaşlıklı konuda konulara bakın. IBM WebSphere MQiçin yazabileceğiniz uygulama tipleriyle ilgili bilgi edinmek için “Uygulamaların geliştirilmesi” sayfa 7konusuna bakın.

Uygulama geliştirilmesine özgü IBM WebSphere MQ kavramlarını öğrenmek için aşağıdaki bağlantıları kullanın:

- “IBM WebSphere MQ ileti” sayfa 9
- Noktadan noktaya ileti alışverişi
- WebSphere MQ yayınlama/abone olma mesajlarına giriş
- “İstemci uygulamasındaki ileti kuyruğu arabiriminin (MQI) kullanılması” sayfa 336
- “ WebSphere MQ' da Web hizmetlerini kullanma” sayfa 907
- “İleti alışverişi kanallarına ilişkin kanal çıkışı programları” sayfa 378
- “Hareket desteği senaryoları” sayfa 39

MQI kullanan uygulamaları çalıştırabilmeniz için bazı IBM WebSphere MQ nesneleri oluşturmanız gerekir. Daha fazla bilgi için “MQI kullanan uygulama programları” sayfa 9 başlıklı konuya bakın.

İlgili kavramlar

“IBM WebSphere MQ uygulamalarını tasarlama” sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

“Örnek WebSphere MQ programları” sayfa 92
Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

“Kuyruğa alma uygulaması yazılıyor” sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

“İstemci uygulamaları yazılıyor” sayfa 335
What you need to know to write client applications on WebSphere MQ.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“JMS için WebSphere MQ sınıflarının kullanılması” sayfa 687
Java Message Service için WebSphere MQ sınıfları (JMS için WebSphere MQ sınıfları), WebSphere MQile birlikte verilen JMS sağlayıcısıdır. javax.jms paketinde tanımlanan arabirimlerin yanı sıra, JMS için WebSphere MQ sınıfları JMS API ' ya iki uzantı kümesi sağlar.

“Component Object Model Interface olanağının kullanılması (ActiveXiçin WebSphere MQ Automation Sınıfları)” sayfa 992
WebSphere MQ Automation Classes for ActiveX (MQAX), uygulamanıza WebSphere MQ' a erişmek için kullanabileceğiniz sınıfları sağlayan ActiveX bileşenleridir.

“Java için WebSphere MQ sınıflarının kullanılması” sayfa 627
Java için WebSphere MQ sınıfları, Java ortamında WebSphere MQ ' yı kullanmanıza olanak sağlar. A Java application can use either WebSphere MQ classes for Java or WebSphere MQ classes for JMS to access WebSphere MQ resources.

“.NET kullanımı” sayfa 537
.NET için WebSphere MQ sınıfları, .NET programlama çerçevesinde yazılmış bir programın WebSphere MQ ' a bir WebSphere MQ MQI istemcisi olarak bağlanmasını ya da doğrudan bir WebSphere MQ Server sunucusuna bağlanmasını sağlar.

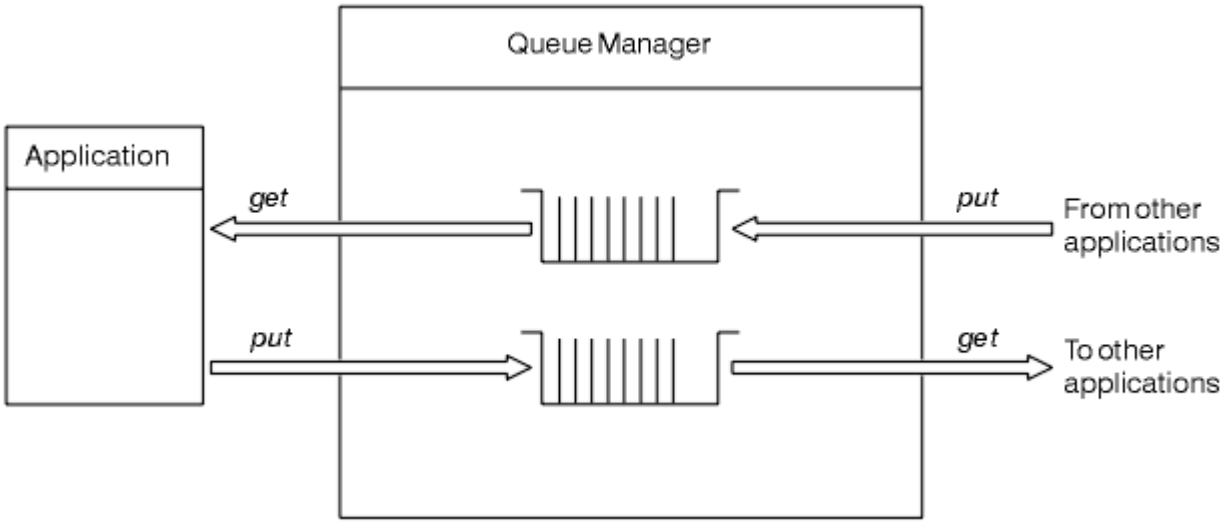
“C++ kullanılması” sayfa 603
WebSphere MQ provides C++ classes equivalent to WebSphere MQ objects and some additional classes equivalent to the array data types. Bu, MQI aracılığıyla olmayan bazı özellikleri sağlar.

“IBM WebSphere MQ uygulaması oluşturulması” sayfa 409
Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

MQI kullanan uygulama programları

IBM WebSphere MQ uygulama programlarının başarıyla çalıştırılabilmesi için bazı nesnelere gereksinim vardır.

Şekil 1 sayfa 9 , bir kuyruktan iletileri kaldıran, bunları işleyen ve daha sonra, aynı kuyruk yöneticisinden başka bir kuyruğa sonuç gönderen bir uygulamayı gösterir.



Şekil 1. Kuyruklar, iletiler ve uygulamalar

Uygulamalar, iletileri yerel ya da uzak kuyruklara (`MQPUT`kullanarak) yerleştirebilse de, bu iletiler yalnızca doğrudan yerel kuyruklardan (`MQGET`kullanılarak) ileti alabilirler.

Bu uygulamanın çalışabilmesi için aşağıdaki koşulların yerine getirilmesi gerekir:

- Kuyruk yöneticisi var olmalı ve çalışıyor olmalıdır.
- İletilerin kaldırılacağı ilk uygulama kuyruğu tanımlanmalıdır.
- Uygulamanın iletileri yerleştirdiği ikinci kuyruğun da tanımlanması gerekir.
- Uygulamanın kuyruk yöneticisine bağlanabilmesi gerekir. Bunu yapmak için IBM WebSphere MQile bağlantı oluşturulmalıdır. Bkz. “IBM WebSphere MQ uygulaması oluşturulması” sayfa 409.
- İletileri ilk sıraya koyan uygulamalar aynı zamanda bir kuyruk yöneticisine bağlanmalıdır. Bunlar uzaksa, iletim kuyrukları ve kanallarıyla da ayarlanmalıdır. Sistemin bu bölümü Şekil 1 sayfa 9içinde gösterilmez.

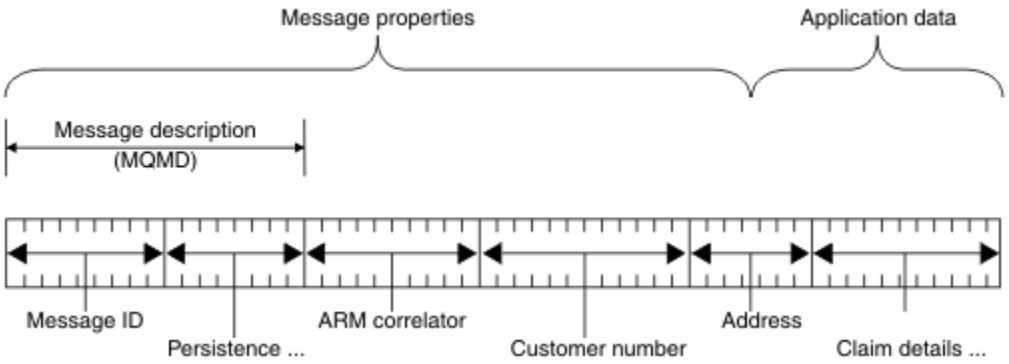
IBM WebSphere MQ ileti

Bu bilgiler, IBM WebSphere MQ ileti kavramını, ileti parçalarını ve ileti tanımlayıcısını tanıtır.

IBM WebSphere MQ iletileri iki bölümden oluşur:

- İleti Özellikleri
- Uygulama Verileri

Şekil 2 sayfa 10 , bir iletiyi gösterir ve ileti özellikleri ve uygulama verilerine mantıksal olarak nasıl bölüneceğini gösterir.



Şekil 2. İletinin gösterimi

Bir WebSphere MQ iletisinde taşınan uygulama verileri, üzerinde veri dönüştürme işlemi gerçekleştirilmediği sürece kuyruk yöneticisi tarafından değiştirilmez. Ayrıca, WebSphere MQ bu verilerin içeriğine herhangi bir kısıtlama koymaz. Her iletteki verilerin uzunluğu hem kuyruğun, hem de kuyruk yöneticisinin *MaxMsgLength* özniteliğinin değerini aşamaz.

On WebSphere MQ for AIX, WebSphere MQ for HP-UX, WebSphere MQ for Linux, WebSphere MQ for Solaris, and WebSphere MQ for Pencereler, the *MaxMsgLength* defaults to 100 MB (104 857 600 bytes).

Bazı durumlarda, iletilerinizi *MaxMsgLength* özniteliğinin değerinden biraz daha kısa hale getiriniz. Daha fazla bilgi için bkz. “İletinizdeki veriler” sayfa 220 .

MQPUT ya da MQPUT1 MQI çağrılarını kullandığınızda bir ileti yaratasınız. Bu çağrılara giriş olarak, denetim bilgilerini (iletinin önceliği ve yanıt kuyruğu adı gibi) ve verilerinizi ve aramayı, iletiyi bir kuyruğa yerleştirdiğinizde sağlardınız. Bu çağrılar hakkında daha fazla bilgi için bkz. MQPUT ve MQPUT1 .

İleti tanımlayıcısı
İleti denetimi bilgilerine, *ileti tanımlayıcısını*tanımlayan MQMD yapısını kullanarak erişebilirsiniz.

MQMD yapısının tam açıklaması için bakınız: MQMD-Message descriptor.

İletinin kökeniyle ilgili bilgileri içeren MQMD içindeki alanların nasıl kullanılacağını görmek için “İleti bağlamı” sayfa 37 ' e bakın.

İleti açıklayıcısının farklı sürümleri var. İletilerin gruplanmasına ve bölümlerine ilişkin ek bilgiler (bkz. “İleti grupları” sayfa 34), ileti tanımlayıcısının (ya da MQMDE) Sürüm 2 'de sağlanır. Bu, Sürüm 1 ileti tanımlayıcısında aynıdır, ancak ek alanlar içerir. Bunlar MQMDE-Message Descriptor uzantısındaaçıklanmıştır.

İleti tipleri

IBM WebSphere MQtarafından tanımlanan dört tip ileti vardır.

Bu dört ileti şunlardır:

- Veri Paketi
- İstek iletileri
- Yanıt iletileri
- İletileri raporla
  - Rapor iletisi tipleri
  - Rapor iletisi seçenekleri

Uygulamalar, kendi aralarında bilgi aktarmak için ilk üç tip ileti tipini kullanabilir. Dördüncü tip, rapor, uygulamalar ve kuyruk yöneticilerinin, bir hatanın ortaya çıkma gibi olaylarla ilgili bilgileri raporlamak için kullanabilmeleri içindir.

Her ileti tipi bir MQMT_ * değeriyle tanıtılır. Kendi ileti tiplerinizi de tanımlayabilirsiniz. Kullanabileceğiniz değerler aralığı için bkz. MsgType.

Veri Paketleri

İletiyi alan uygulamadan (yani, iletiyi kuyruktan alır) yanıtlamak zorunda kalmadığınızda bir *veri paketi* kullanın.

Veri paketlerini kullanabilecek bir uygulama örneği, bir havaalanı salonunda uçuş bilgilerini görüntüleyen bir uygulamadır. Bir ileti, tüm uçuş bilgileri ekranının verilerini içerebilir. Bir iletinin teslim edilmemesi büyük olasılıkla önemli olmadığı için, böyle bir uygulamanın ileti için bir onay isteme olasılığı düşük. Uygulama kısa bir süre sonra bir güncelleme iletisi gönderir.

İstek iletileri

İletiyi alan uygulamadan yanıt almak istediğinizde bir *istek iletisi* kullanın.

İstek iletilerini kullanabilecek bir uygulama örneği, bir denetleme hesabının bakiyesini gösteren bir uygulamadır. İstek iletisi hesap numarasını içerebilir ve yanıt iletisi hesap bakiyesini içerir.

Yanıt iletinizi istek iletinizle bağlantılamak istiyorsanız, iki seçenek vardır:

- İstek iletisine, istek iletisine ilişkin bilgi koymasını sağlamaktan sorumlu istek iletisini işleyen uygulamayı yapın.
- Yanıt iletisinin *MsgId* ve *CorrelId* alanlarının içeriğini belirtmek için istek iletinizin ileti tanımlayıcısındaki rapor alanını kullanın.
  - Özgün iletinin *MsgId* ya da `CorrelId` iletisinin, yanıt iletisinin *CorrelId* alanına kopyalanabileceğini (varsayılan işlemin *MsgId*kopyasıdır) isteyebilirsiniz.
  - Yanıt iletisi için yeni bir `MsgId` oluşturulduğunu ya da özgün iletinin *MsgId* ' in yanıt iletisinin *MsgId* alanına kopyalanabileceğini (varsayılan işlem yeni bir ileti tanıtıcısı oluşturmasıdır) isteyebilirsiniz.

Yanıt iletileri

Başka bir iletiyi yanıtladığınızda bir *yanıt iletisi* kullanın.

Bir yanıt iletisi oluşturduğunuzda, yanıtlamakta olduğunuz iletinin ileti tanımlayıcısında ayarlanan tüm seçeneklere saygı göstermeniz gerekir. Rapor seçenekleri, ileti tanıtıcısı (`MsgId`) ve ilinti tanıtıcısı (`CorrelId`) alanlarının içeriğini belirtir. Bu alanlar, yanıtı alan uygulamanın, yanıtı özgün isteğiyle ilintilendirmesine olanak tanır.

Rapor iletileri

Rapor iletileri , bir iletiyi işlerken oluşan bir hatanın ortaya çıkma gibi olaylar hakkında uygulamaları bilgilendirir.

Bu bilgiler aşağıdaki tarafından oluşturulabilir:

- Kuyruk yöneticisi,
- Bir ileti kanalı aracısı (örneğin, iletiyi teslim edemezse) ya da
- Bir uygulama (örneğin, iletide verileri kullanamazsa).

Rapor iletileri herhangi bir zamanda oluşturulabilir ve uygulamanız bunları beklemediğinde bir kuyruğa gelebilir.

Rapor iletisi türleri

Bir kuyruğa ileti yerleştirdiğinizde, aşağıdaki bilgileri almayı seçebilirsiniz:

- *Kural dışı durum raporu iletisi*. Bu, kural dışı durum işareti ayarına sahip bir iletiye yanıt olarak gönderilir. İleti kanalı aracısı (MCA) ya da uygulama tarafından üretilir.

- Bir *süre bitimi rapor iletisi*. Bu, bir uygulamanın süre sonu eşiğine ulaşmış bir iletiyi almayı denediğini; ileti atılır olarak işaretlendi. Bu rapor tipi, kuyruk yöneticisi tarafından oluşturulur.
- Bir *geliş onayı (COA) rapor iletisi*. Bu, iletinin hedef kuyruğuna ulaştığını gösterir. Kuyruk yöneticisi tarafından üretilir.
- *Teslim edilme (COD) rapor iletisi*. Bu, iletinin alan bir uygulama tarafından alındığını gösterir. Kuyruk yöneticisi tarafından üretilir.
- *Pozitif işlem bildirimi (PAN) rapor iletisi*. Bu, bir isteğin başarıyla hizmet verdiğine (yani, iletide istenen işlemin başarıyla gerçekleştirildiğini) gösterir. Bu rapor tipi uygulama tarafından oluşturulur.
- *Negatif eylem bildirimi (NAN) rapor iletisi*. Bu, bir isteğin başarıyla gerçekleştirilmediğini gösterir (yani, iletide istenen işlem başarıyla gerçekleştirilmedi). Bu rapor tipi uygulama tarafından oluşturulur.

Not: Her rapor iletisi tipi aşağıdakilerden birini içerir:

- Özgün iletinin tamamı
- Özgün iletteki ilk 100 byte veri
- Özgün iletiden veri yok

Bir kuyruğa ileti yerleştirdiğinizde birden çok rapor iletisi tipi isteyebilirsiniz. Teslim onayı rapor iletisini ve kural dışı durum raporu ileti seçeneklerini seçerseniz, ileti teslim edilmezse, bir kural dışı durum raporu alırsınız. Ancak, yalnızca teslim onayı rapor iletisi seçeneğini belirlerseniz ve ileti teslim edilmezse, bir kural dışı durum raporu iletisi almayın.

Belirli bir iletiyi oluşturmaya ilişkin ölçütler karşılandığında, sizin istediğiniz rapor iletileri, yalnızca sizin aldığınız iletilerdir.

Rapor iletisi seçenekleri

Bir kural dışı durum arisen olduktan sonra bir iletiyi *atabilirsiniz* . Atma seçeneğini belirlerseniz ve bir kural dışı durum raporu iletisi istediyseniz, rapor iletisi *ReplyToQ* ve *ReplyToQMgr*' e gider ve özgün ileti atılır.

Not: Bunun bir yararı, ölü mektup kuyruğuna giden ileti sayısını azaltabilirsiniz. Ancak, yalnızca veri paketi iletileri göndermediği sürece başvurunuzun döndürülen iletilerle uğraşmak zorunda olduğu anlamına gelir. Bir kural dışı durum raporu iletisi oluşturulduğunda, özgün iletinin kalıcı olarak kalıcılığını devralır.

Bir rapor iletisi teslim edilemezse (kuyruk dolduysa, örneğin), rapor iletisi ölü harf kuyruğunda yerleştirilir.

Bir rapor iletisi almak istiyorsanız, *ReplyToQ* alanında yanıtınızın adını belirtin; tersi durumda, özgün iletinizin MQPUT ya da MQPUT1 MQRC_MISSING_REPLY_TO_Q ile başarısız olur.

İleti için oluşturulan rapor iletilerinin *MsgId* ve *CorrelId* alanlarının içeriğini belirtmek için bir iletinin ileti tanımlayıcısında (MQMD) diğer rapor seçeneklerini de kullanabilirsiniz:

- Özgün iletinin *MsgId* ya da *CorrelId* ' inin rapor iletisinin *CorrelId* alanına kopyalanması için istekte bulunyabilirsiniz. Varsayılan işlem, ileti tanıtıcısını kopyalamandır. MQRO_COPY_MSG_ID_TO_CORRELID iletisini kullanın; bu ileti, yanıtı ya da rapor iletisini özgün iletiyle ilintilendirmek için ileti göndericisini etkinleştirir. Yanıtın ya da rapor iletisinin ilinti tanıtıcısı, özgün iletinin ileti tanıtıcısıyla aynı.
- Rapor iletisi için yeni bir *MsgId* oluşturulduğunu ya da özgün iletinin *MsgId* ' inin rapor iletisinin *MsgId* alanına kopyalanabileceğini isteyebilirsiniz. Varsayılan işlem, yeni bir ileti tanıtıcısı oluşturmandır. MQRO_NEW_MSG_ID ' yi kullanın; sistemdeki her iletinin farklı bir ileti tanıtıcısı olmasını güvenceye alır ve sistemdeki diğer tüm iletilerden ayırt edilemez bir şekilde ayırt edilebilir.
- Özel uygulamaların MQRO_PASS_MSG_ID ya da MQRO_PASS_COREL_ID kullanması gerekebilir. Ancak, kuyruktan iletileri okuyan uygulamayı, örneğin, kuyruğun aynı ileti tanıtıcısına sahip birden çok ileti içerdiğini doğrulamak için kuyruktan okuyan bir uygulamayı tasarlamanız gerekir.

  Sunucu uygulamaları, istek iletisinde bu işaretlerin ayarlarını denetlemeli ve yanıt ya da rapor iletisinde *MsgId* ve *CorrelId* alanlarını uygun şekilde ayarlamalıdır.

İstekte bulunanın uygulaması ile sunucu uygulaması arasında aracı olarak işlev görmekte olan uygulamalar, bu işaretlerin ayarlarını denetlemesine gerek yoktur. This is because these applications typically need to forward the message to the server application with the *MsgId*, *CorrelId*, and *Report* fields unchanged. Bu, sunucu uygulamasının, yanıt iletisinin *CorrelId* alanındaki özgün iletiden *MsgId* dosyasını kopyalamasına olanak sağlar.

Bir iletiyle ilgili rapor oluştururken, sunucu uygulamalarının bu seçeneklerden herhangi birinin ayarlanmış olup olmadığını test etmesi gerekir.

Rapor iletilerinin nasıl kullanılacağı hakkında daha fazla bilgi için Rapor' a bakın.

Raporun niteliyi belirtmek için, kuyruk yöneticileri bir dizi geribildirim kodu kullanır. Bu kodları, bir rapor iletisinin ileti tanımlayıcısının *Feedback* alanına koydular. Queue managers can also return MQI reason codes in the *Feedback* field. IBM WebSphere MQ Uygulamaların kullanması için bir geribildirim kodu aralığı tanımlar.

Geribildirim ve neden kodlarıyla ilgili daha fazla bilgi için bkz. Feedback.

Geribildirim kodu kullanabilecek bir programa örnek olarak, kuyruğa hizmet eden diğer programların iş yüklerini izleyen bir program örneği. Kuyruğa gönderme yapan bir programın birden çok eşgörünümü varsa ve kuyruğa gelen iletilerin sayısı artık bu durumu haklı çıkarmadıysa, bu tür bir program, programın etkinliğini sonlandırması gerektiğini belirtmek üzere hizmet veren programlardan birine (geri bildirim kodu MQFB_QUIT ile) bir rapor iletisi gönderebilirler. (Bir izleme programı, bir kuyruğa kaç program hizmet verdiğini öğrenmek için MQINQ çağrısını kullanabilir.)

Raporlar ve kesimlere ayrılmış iletiler

Not supported on WebSphere MQ for z/OS .

Bir ileti bölümlendiyse (kesimlere ayrılmış iletilerin açıklaması için bkz. “İleti bölümlemesi” sayfa 251) ve rapor oluşturulmasını istediğinizde, iletinin bölümlenmemesinden daha fazla rapor alabilirsiniz.

WebSphere MQtarafından oluşturulan raporlar için

İletilerinizi bölümlediyseniz ya da kuyruk yöneticisinin bunu yapmasını izin verdiyseniz, iletinin tamamı için tek bir rapor almayı bekleyebileceğiniz tek bir vaka vardır. Bu, yalnızca COD raporlarını istediğinizde ve uygulama alma işlemi sırasında MQGMO_COMPLETE_MSG belirtiminiz yer alıyor.

Diğer durumlarda, uygulamanızın çeşitli raporlarla başa çıkabilmek için hazırlanması gerekir; genellikle her bir kesim için bir tane olmalıdır.

Not: İletilerinizi bölümlediyseniz ve döndürülebilmek için özgün ileti verilerinin yalnızca ilk 100 baytına gereksinim duyarsanız, 100 ya da daha fazla göreli konumu olan kesimler için veri olmadan rapor isteyecek rapor seçeneklerinin ayarını değiştirin. Bunu yapmazsanız ve her bir kesim 100 baytlık veri istemesi için ayarı bırakırsanız ve rapor iletilerini tek bir MQGET ile, MQGMO_COMPLETE_MSG belirtilerek alırsınız. Raporlar, her uygun göreli konumda 100 baytlık okuma verisi içeren büyük bir iletiye toplanma yapar. Bu durumda, büyük bir arabelleğe gereksinim duyarsınız ya da MQGMO_ACCEPT_TRUNCATED_MSG belirtmeniz gerekir.

Uygulamalar tarafından oluşturulan raporlar için

Uygulamanız raporlar oluşturduysa, her zaman özgün ileti verilerinin başlangıcındaki mevcut olan WebSphere MQ üstbilgilerini rapor iletisi verisine kopyalayın.

Daha sonra, rapor iletisi verilerine özgün ileti verilerinin (ya da genellikle içereceği diğer miktarı) 100 byte 'ı ya da tümünü ekleyin.

MQMD ' den başlayarak ve var olan tüm üstbilgiler arasında devam ederek, ardışık biçim adlarına bakarak kopyalanması gereken WebSphere MQ üstbilgilerini tanıyabilirsiniz. Aşağıdaki `Format` adları şu WebSphere MQ üstbilgilerini gösterir:

- MQMDE
- MQDLH

- MQXQH
- MQIH
- MQH*

MQH*, MQH karakterleriyle başlayan herhangi bir ad anlamına gelir.

`Format` adı, MQDLH ve MQXQH için belirli konumlarda gerçekleşir, ancak diğer WebSphere MQ üstbilgileri için aynı konumda oluşur. Üstbilginin uzunluğu, MQMDE, MQIMS ve tüm MQH* üstbilgileri için aynı konumda bulunan bir alanda bulunuyor.

Bir Sürüm 1 MQMD kullanıyorsanız ve bir kesime ya da bir gruptaki bir iletiye ya da bölümlemeye izin verilmeye izin verilen bir iletiye bildiriyorsanız, rapor verileri bir MQMDE ile başlamalıdır. Set the *`OriginalLength`* field to the length of the original message data excluding the lengths of any WebSphere MQ headers that you find.

Raporlar alınıyor

COA ya da COD raporları için, MQGMO_COMPLETE_MSG ile sizin için yeniden derlemelerini isteyebilirsiniz.

Bir MQGMO_COMPLETE_MSG içeren bir MQGET, yeterli sayıda rapor iletisi (tek tip, örneğin COA ve aynı *`GroupId`*ile birlikte), tek bir tam özgün iletiyi göstermek için kuyruğunda hazır olduğunda memnun olur. Bu, rapor iletileri tamamen özgün verileri içermese de geçerlidir; her bir rapor iletisinde *`OriginalLength`* alanı, verinin kendisi mevcut olmasa bile, o rapor iletiyle temsil edilen özgün verilerin uzunluğunu verir.

Bu tekniği, kuyruğunda (örneğin, hem COA hem de COD gibi) birkaç farklı rapor tipi olsa da, MQGMO_COMPLETE_MSG ile bir MQGET işlemi yalnızca aynı *`Feedback`* kodlarına sahip olduğunda iletileri yeniden birleştirdiği için kullanabilirsiniz. Ancak, genellikle bu tekniği kural dışı durum raporları için kullanamazsınız; çünkü, genel olarak, bunlar farklı *`Feedback`* kodlarına sahiptir.

Bu tekniği, iletinin tamamının geldiğine ilişkin olumlu bir gösterge elde etmek için kullanabilirsiniz. Ancak, bazı kesimlerin başkaları tarafından bir özel durum (ya da süre bitimi) oluşturabilirken bazı kesimlerin gelmesi olasılığına karşı en çok ihtiyaç duyarsanız, bu durumda bir süre daha gerekir. Bu durumda MQGMO_COMPLE_MSG kullanamazsınız; genel olarak, farklı bölümler için farklı *`Feedback`* kodları alabilirsiniz ve bir bölüm için birden çok rapor alabilirsiniz. Ancak, MQGMO_ALL_SEGMENTS_AVALABILIR seçeneğini kullanabilirsiniz.

Bunun için izin vermek için raporları geldikleri gibi almanız ve özgün iletiye ne olduğu ile ilgili uygulamanıza bir resim oluşturmanız gerekebilir. Rapor iletisinde *`GroupId`* alanını, özgün iletinin*`GroupId`* ile ilintilendirmek için rapor iletisinde ve her bir rapor iletisinin tipini tanımlamak için *`Feedback`* alanını kullanabilirsiniz. Bunu yapma şekliniz, uygulama gereksinimlerinize bağlıdır.

Bir yaklaşım aşağıdaki gibidir:

- COD raporları ve kural dışı durum raporları isteyin.
- Belirli bir süreden sonra, MQGMO_COMPLETE_MSG kullanılarak bir COD raporlarının eksiksiz bir kümesinin alınıp alınmadığını denetleyin. Böyle bir durumda, uygulamanız iletinin tamamının işlendiğini bilir.
- Bu ileti yoksa ve bu iletiyle ilgili kural dışı durum raporları varsa, sorunu bölümlere ayrılmamış iletiler için kullanın, ancak bir noktada artık artık bölümleri temizlediğinizden emin olun.
- Herhangi bir türde rapor olmayan bölümler varsa, özgün kesimler (ya da raporlar), bir kanalın yeniden bağlanmasını bekliyor olabilir ya da ağ bir noktada aşırı yüklenmiş olabilir. Herhangi bir kural dışı durum raporu alınmadıysa (ya da yalnızca geçici olduğunu düşünüyorsanız), uygulamanızın biraz daha beklemesine izin verebileceğinizi düşünebilirsiniz.

  Daha önce olduğu gibi, bu durum, bölümlenmemiş iletiler ile çalışırken sahip olduğunuz dikkate alınacak hususlara benzer; ancak, artık yetim kesimlerinin temizlenme olasılığını da göz önünde bulundurmanız gerekir.

Özgün ileti kritik değilse (örneğin, bir sorguysa ya da daha sonra yinelenebilecek bir iletiyse), artık kesimlerin kaldırılmasını sağlamak için bir süre bitimi ayarlayın.

Arka düzey kuyruk yöneticileri

Bir rapor, bölümlemeyi destekleyen bir kuyruk yöneticisi tarafından oluşturulduğunda, ancak *değil* destek bölümlemesini destekleyen bir kuyruk yöneticisinde alındığında, MQMDE yapısı (reportile temsil edilen *Offset* ve *OriginalLength* ' yi tanımlar) rapor verilerine her zaman sıfır, 100 bayt ya da iletteki özgün verilerin yanı sıra eklenir.

Ancak, bir iletinin bir bölümü, segmentasyonu desteklemeyen bir kuyruk yöneticisinde geçerse, orada bir rapor oluşturulduysa, özgün iletteki MQMDE yapısı tamamen veri olarak işlenir. Bu nedenle, özgün verilerin sıfır byte 'ı istendiyse, rapor verisine dahil edilmez. MQMDE olmadan, rapor iletisi yararlı olmayabilir.

Bir iletinin arka düzey kuyruk yöneticisi aracılığıyla seyahat edebileceğinden emin olmak için, raporlarda en az 100 bayt veri isteyin.

İleti denetim bilgileri ve ileti verileri biçimi

Kuyruk yöneticisi yalnızca bir ileti içindeki denetim bilgilerinin biçimiyle ilgilenirken, iletiyi işleyen uygulamalar hem denetim bilgilerinin, hem de verilerin biçimiyle ilgilenir.

İleti denetim bilgilerinin biçimi

İleti tanımlayıcısının karakter dizilimi alanlarındaki denetim bilgilerinin, kuyruk yöneticisi tarafından kullanılan karakter kümesinde olması gerekir.

Kuyruk yöneticisi nesnesinin *CodedCharSetId* özniteliği bu karakter kümesini tanımlar. Denetim bilgileri bu karakter kümesinde olmalıdır; çünkü, uygulamalar bir kuyruk yöneticisinden diğerine iletileri aktarırken, iletileri ileten ileti kanalı aracıları, hangi veri dönüştürmenin gerçekleştirileceğini saptamak için bu özniteliğin değerini kullanır.

İleti verilerinin biçimi

Aşağıdakilerden herhangi birini belirleyebilirsiniz:

- Uygulama verilerinin biçimi
- Karakter verilerinin karakter takımı
- Sayısal verilerin biçimi

Bunu yapmak için şu alanları kullanın:

Format
Bir iletinin alıcısına, iletindeki uygulama verilerinin biçiminin bir ileti olduğunu gösterir.

Kuyruk yöneticisi bir ileti yarattığında, bazı durumlarda o iletinin biçimini tanımlamak için *Format* alanını kullanır. Örneğin, bir kuyruk yöneticisi iletiyi teslim edemediğinde, iletiyi ölüme mektup (teslim edilmemiş ileti) kuyruğuna koyar. İletiye bir üstbilgi ekler (daha fazla denetim bilgisi içerir) ve bunu göstermek için *Format* alanını değiştirir.

Kuyruk yöneticisinin adları MQ(örneğin, MQFMT_STRING) ile başlayan *yerleşik biçimler* sayısı. Bunlar gereksinimlerinizi karşılamazsa, kendi biçimlerinizi (*kullanıcı tanımlı biçimler*) tanımlayabilir, ancak bunlar için MQ ile başlayan adları kullanmamalısınız.

Kendi biçimlerinizi yarattığınızda ve kullandığınızda, iletiyi MQGMO_CONVERT kullanarak almak için bir veri dönüştürme çıkışı yazmanız gerekir.

CodedCharSetId
Bu, iletteki karakter verilerinin karakter kümesini tanımlar. Bu karakter kümesini kuyruk yöneticisininde ayarlamak istiyorsanız, bu alanı MQCCSI_Q_MGR ya da MQCCSI_INHERIT değişmezine ayarlayabilirsiniz.

Bir kuyruktan ileti aldığınızda, *CodedCharSetId* alanının değerini, uygulamanızın beklediği değerle karşılaştırın. İki değer farklı olursa, iletteki herhangi bir karakter verilerini dönüştürmeniz ya da bir veri dönüştürme iletisi çıkışı kullanmanız gerekebilir.

Encoding

Bu, ikili tamsayıları, paketlenmiş ondalık tamsayıları ve kayan nokta numaralarını içeren sayısal ileti verilerinin biçimini açıklar. Genellikle, kuyruk yöneticisinin üzerinde çalıştığı belirli bir makineye göre kodlanır.

Bir kuyruğa ileti koyduğunuzda, genellikle *Encoding* alanında MQKEN_NATIVE değişmezini belirtiyorsunuz. Bu, ileti verilerinizin kodlamasının, uygulamanızın çalıştığı makineniyle aynı olduğu anlamına gelir.

Bir kuyruktan ileti aldığınızda, ileti tanımlayıcısındaki *Encoding* alanının değerini, makinenizdeki sabit MQENC_NATIVE değişmezinin değeriyle karşılaştırın. İki değer farklı olursa, iletteki sayısal verileri dönüştürmeniz ya da varsa, veri dönüştürme iletisi çıkışı kullanmanız gerekebilir.

Uygulama verileri dönüştürme

Uygulama verilerinin karakter kümesine dönüştürülmesine ve farklı platformların ilgilendiği başka bir uygulama için gereken kodlamaya dönüştürülmesi gerekebilir.

Bu değer, gönderme kuyruğu yöneticisinde ya da alıcı kuyruk yöneticisinde dönüştürülebilmektedir. Yerleşik biçimlerin kitaplığı gereksinimlerinizi karşılamazsa, kendi biçiminizi tanımlayabilirsiniz. Dönüştürme tipi, ileti tanımlayıcısının biçim alanında, MQMD ' de belirtilen ileti biçimine bağlıdır.

Not: Belirtilen MQFMT_NONE ile iletiler dönüştürülmedi.

Gönderme kuyruğu yöneticisinde dönüştürme

Uygulama verilerini dönüştürmek için gönderilen ileti kanalı aracısına (MCA) gerek duyarsanız, CONVERT kanalı özniteliğini YES olarak ayarlayın.

Dönüştürme, belirli yerleşik biçimler için gönderme kuyruğu yöneticisinde ve uygun bir kullanıcı çıkışı sağlandıysa, kullanıcı tanımlı biçimler için gerçekleştirilir.

Yerleşik biçimler

Bu üyeler şunlardır:

- Tüm karakterler (MQFMT_STRING biçim adını kullanarak) olan iletiler
- WebSphere MQ tanımlı iletiler (Programlanır Komut Biçimleri gibi)

  WebSphere MQ , yönetim iletileri ve olayları için Programlanır Komut Biçimi iletilerini kullanır (bu durumda kullanılan biçim adı, MQFMT_ADMIN 'dir). Kendi iletileriniz için aynı biçimi (MQFMT_PCF biçim adını kullanarak) kullanabilir ve yerleşik veri dönüştürmeden yararlanabilirsiniz.

Kuyruk yöneticisi yerleşik biçimlerinin tümünün MQFMT ile başlayan adları var. Bunlar listelenir ve Biçimalanında açıklanmaktadır.

Uygulama tanımlı biçimler

Kullanıcı tanımlı biçimler için, uygulama verileri dönüştürmesi bir veri dönüştürme çıkış programı tarafından gerçekleştirilmelidir (daha fazla bilgi için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397). İstemci-sunucu ortamında, çıkış sunucuya yüklenir ve dönüştürme işlemi burada gerçekleşir.

Alıcı kuyruk yöneticisinde dönüştürme

Uygulama iletisi verileri, hem yerleşik hem de kullanıcı tanımlı biçimler için alıcı kuyruk yöneticisi tarafından dönüştürülebilmektedir.

MQGMO_CONVERT seçeneğini belirtirseniz, dönüştürme işlemi bir MQGET çağrısının işlenmesi sırasında gerçekleştirilir. Ayrıntılar için Seçenekler' e bakın.

Kodlanmış karakter takımları

WebSphere MQ ürünleri, temeldeki işletim sistemi tarafından sağlanan kodlanmış karakter takılarını destekler.

Bir kuyruk yöneticisi yarattığınızda, kullanılan kuyruk yöneticisi kodlanmış karakter takımı tanıtıcısı (CCSID), temeldeki ortamın temelini temel alır. Bu bir karma kod sayfasıysa, WebSphere MQ karma kod sayfasının SBCS kısmını kuyruk yöneticisi CCSID 'si olarak kullanır.

Genel veri dönüştürmesi için, temeldeki işletim sistemi DBCS kod sayfalarını destekliyorsa, WebSphere MQ bunu kullanabilir.

Desteklediği kodlanmış karakter kümelerinin ayrıntıları için işletim sisteminize ilişkin belgelere bakın.

Birden çok platforma yayılan uygulamalar yazarken uygulama verileri dönüştürme, biçim adları ve kullanıcı çıkışlarını dikkate almanız gerekir. Veri dönüştürme çıkışlarının çağrılmasına ve yazılmasına ilişkin bilgi için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397 .

İleti öncelikleri

İletiyi bir kuyruğa koyduğunuzda, bir iletinin önceliğini (MQMD yapısının *Priority* alanında) ayarlardınız. Öncelik için bir sayısal değer ayarlayabilir ya da iletinin, kuyruğun varsayılan önceliğini almasına izin verirsiniz.

Kuyruğun *MsgDeliverySequence* özniteliği, kuyrukta bulunan iletilerin FIFO (ilk giren, ilk çıkış) sırasıyla ya da öncelik sırası içinde FIFO ' da saklanıp saklanmayacağını belirler. Bu öznitelik MQMDS_PRIORITY değerine ayarlıysa, iletiler ileti tanımlayıcılarının *Priority* alanında belirtilen önceliğe göre kuyruğa alınır; ancak, bu, MQMDS_FIFO olarak ayarlandıysa, iletiler kuyruğun varsayılan önceliğiyle kuyruğa alınır. Eşit önceliğe sahip iletiler geliş sırasına göre kuyrukta saklanır.

Bir kuyruğun *DefPriority* özniteliği, kuyruğa konmakta olan iletiler için varsayılan öncelik değerini ayarlar. Bu değer, kuyruk yaratıldığında ayarlanır, ancak daha sonra değiştirilebilir. Diğer ad kuyrukları ve uzak kuyruklara ilişkin yerel tanımlamalar, çözümledikleri temel kuyruklardan farklı varsayılan önceliklere sahip olabilir. Çözüm yolunda birden çok kuyruk tanımlaması varsa (bkz. “Ad çözünürlüğü” sayfa 208), varsayılan öncelik, açık komutta belirtilen kuyruğun *DefPriority* özniteliğinin değerinden (put işleminin sırasında) alınır.

Kuyruk yöneticisinin *MaxPriority* özniteliğinin değeri, o kuyruk yöneticisi tarafından işlenen bir iletiye atayabileceğiniz en yüksek önceliğidir. Bu özniteliğin değerini değiştiremezsiniz. WebSphere MQ' da, öznitelik 9 değerine sahiptir; 0 (en düşük) ile 9 (en yüksek) arasında öncelikler içeren iletiler oluşturabilirsiniz.

İleti Özellikleri

Bir uygulamanın işlenecek iletileri seçmesine izin vermek ya da MQMD ya da MQRFH2 üstbilgilerine erişmeden bir iletiyle ilgili bilgileri almak için ileti özelliklerini kullanın. Bunlar, WebSphere MQ ile JMS uygulamaları arasında iletişimi de kolaylaştırır.

İleti özelliği, metinli bir ad ve belirli bir tipteki bir değerden oluşan bir iletiyle ilişkilendirilmiş verilerdir. İleti özellikleri, yayınların konulara süzmesi ya da kuyruktan seçmeli olarak ileti almak için ileti seçicileri tarafından kullanılır. İleti özellikleri, iş verilerini ya da durum bilgilerini uygulama verilerinde saklamaya gerek kalmadan içermek için kullanılabilir. Bu veri yapılarındaki alanlar, Message Queue Interface (MQI) işlev çağrıları kullanılarak ileti özellikleri olarak erişilebildiğinden, uygulamaların MQ Message Descriptor (MQMD) ya da MQRFH2 üstbilgilerindeki verilere erişmek zorunda kalmaması gerekir.

The use of message properties in WebSphere MQ mimics the use of properties in JMS. Bu, bir JMS uygulamasındaki özellikleri ayarlayabileceğiniz ve bunları bir yordamsal WebSphere MQ uygulamasında ya da başka bir şekilde alabileceğiniz anlamına gelir. Bir özelliği bir JMS uygulaması için kullanılabilir yapmak için, "usr" önekine atayın; daha sonra, bir JMS iletisi kullanıcı özelliği olarak kullanılabilir (önek olmadan). For example, the WebSphere MQ property *usr.myproperty* (a character string) is accessible to a JMS application using the JMS call `message.getStringProperty('myproperty')`. JMS uygulamalarının, iki ya da daha fazla U+002E (".") içerirse "usr" önekine sahip özelliklere erişemediğini unutmayın. karakterdir. Öneki olmayan ve U+002E olmayan bir özellik (".") karakter, "usr" önekine sahip olduğu gibi işlem görür. Bunun tersine, bir JMS uygulamasına "usr" eklenerek bir WebSphere MQ uygulamasında ayarlanmış bir kullanıcı özelliği kümesine erişilebilir. Bir MQINQMP çağrısında sorgulanacak özellik adına önek ekleyin.

İleti özellikleri ve ileti uzunluğu

Bir WebSphere MQ kuyruk yöneticisinde herhangi bir iletiyle akabilecek özelliklerin büyüklüğünü denetlemek için kuyruk yöneticisi özniteliğini *MaxPropertiesLength* kullanın.

Genel olarak, özellikleri ayarlamak için MQSETMP kullandığınızda, özelliğin büyüklüğü, byte cinsinden özellik adının uzunluğunun yanı sıra, MQSETMP çağrısına aktarılan byte cinsinden özellik değerinin uzunluğunu da içerir. Bu, özelliğin karakter kümesi ve özelliğin Unicode 'a dönüştürülebilmesi nedeniyle hedefe iletilmesi sırasında değişebilir; bu durumda, özelliğin büyüklüğü değişebilir.

Bir MQPUT ya da MQPUT1 çağrısında, iletinin özellikleri, kuyruk ve kuyruk yöneticisi için iletinin uzunluğuna doğru sayılmaz, ancak kuyruk yöneticisi tarafından algılanan özelliklerin uzunluğuna kadar (bu iletiler, MQI çağrıları kullanılarak ayarlanmış ya da değil), özelliklerin uzunluğuna kadar sayı sayır.

Özelliklerin büyüklüğü özellik uzunluğu üst sınırını aşarsa, bu ileti MQRC_PROTETIES_TOO_BüFK ile reddedilir. Özelliklerin boyutu, gösterimine bağlı olduğundan, özellik uzunluğu üst sınırını bir brüt düzeyde ayarlamalısınız.

Arabellek özellikleri içeriyorsa, bir uygulamanın, *MaxMsgUzunluğu*değerinden daha büyük bir arabelleğe sahip bir iletiyi başarıyla yerleştirmesi mümkündür. Bunun nedeni, MQRFH2 öğeleri olarak gösterilse bile, ileti özellikleri iletinin uzunluğuna doğru sayılmaz. MQRFH2 üstbilgi alanları, özellikler uzunluğuna yalnızca bir ya da daha çok klasör varsa ve üstbilgideki her klasörde özellikler içeriyorsa, bu alanlara veri eklenir. Bir ya da daha çok klasör MQRFH2 üstbilgisinde bulunuyorsa ve herhangi bir klasör özellik içermiyorsa, MQRFH2 üstbilgi alanları ileti uzunluğuna doğru sayılır.

Bir MQGET çağrısında, iletinin özellikleri, ileti uzunluğuna kadar, kuyruk ve kuyruk yöneticisi tarafından dikkate almaz. Ancak, özellikler ayrı olarak sayıldığından, bir MQGET çağrısının döndürdüğü arabelleğin *MaxMsgUzunluğu* özniteliğinin değerinden büyük olması mümkündür.

Uygulamalarınız *MaxMsgUzunluğu* değerini sorgulamaz ve MQGET ' ı çağırmadan önce bu büyüklükte bir arabellek ayıramaz; bunun yerine, yeterince büyük değerlendirdiğiniz bir arabellek ayırın. If the MQGET fails, allocate a buffer guided by the size of the *DataLength* parameter.

MQGET çağrısının *DataLength* değiştirgesi, uygulama verilerinin bayt cinsinden uzunluğunu döndürür ve MQGMO yapısında bir ileti tanıtıcısı belirtilmediyse, sağladığınız arabelleğde döndürülen özellikler döndürülür.

MQPUT çağrısının *Arabellek* değiştirgesi, gönderilecek uygulama iletisi verilerini ve ileti verilerinde gösterilen tüm özellikleri içerir.

Ürünün 7.0 sürümünden önceki bir kuyruk yöneticisine veri akışı sırasında, ileti tanımlayıcısında bulunan iletinin özellikleri dışında, iletinin uzunluğuna iletinin uzunluğuna bakın. Therefore, you should either raise the value of the *MaxMsgUzunluğu* attribute of channels going to a system earlier than Version 7.0 as necessary, to compensate for the fact that more data might be sent for each message. Diğer bir seçenek olarak, kuyruk ya da kuyruk yöneticisi *MaxMsgLength*değerini azaltabilir; böylece, sistemdeki genel veri düzeyi verileri aynı kalır.

İleti tanımlayıcısı ya da her ileti için uzantı dışında, ileti özellikleri için 100 MB ' lik bir uzunluk sınırı vardır.

İç gösteriminde bir özelliğin büyüklüğü, adın uzunluğunun yanı sıra, değerinin büyüklüğü artı özellik için bazı denetim verilerini içerir. İletiye bir özellik eklendikten sonra özellikler kümesi için bazı denetim verileri de vardır.

Özellik adları

Özellik adı bir karakter dizgisidir. Belirli kısıtlamalar, uzunluğuna ve kullanılabilecek karakter kümesine uygulanır.

Özellik adı, bağlamla sınırlı olmadığı sürece, +4095 karakterle sınırlı olmak üzere, büyük/küçük harfe duyarlı bir karakter dizilimidir. Bu sınır, MQ_MAX_PROPERTY_NAME_LENGTH sabiti içinde yer alır.

Bir ileti özelliği MQI çağrısı kullanırken bu uzunluk üst sınırını aştığınızda, çağrı neden kodu MQRC_PROPERTY_NAME_LENGTH_ERR ile başarısız olur.

JMS ' de özellik adı uzunluğu üst sınırı olmadığı için, bir JMS uygulamasının, MQRFH2 yapısında saklanırken geçerli bir WebSphere MQ özellik adı olmayan geçerli bir JMS özelliği adı ayarlamasıdır.

Bu durumda, ayrıştırıldığında, özellik adının yalnızca ilk 4095 karakteri kullanılır; aşağıdaki karakterler kesilir. Bu, seçicileri kullanan bir uygulamanın bir seçim dizgisiyle eşleşmemesi ya da birden çok özelliğin aynı adla kesilebileceğinden, beklemediği bir dizgiyle eşleşmesine neden olabilir. Bir özellik adı kısaltıldığında, WebSphereMQ bir hata günlüğü iletisi yayınlar.

Tüm özellik adları, Java tanıtıcıları için Java Language Specification (Java Dil Belirtimi) ile tanımlanan kuralları izlemelidir; bu kural dışı durum, Unicode karakterinin U+002E (.) adının bir parçası olarak kullanılmasına izin verilir; ancak, bu kural dışı durum başlangıçtan değil, adı altında değil. Java Tanıtıcılarına ilişkin kurallar, özellik adlarına ilişkin JMS belirtiminde içerilenlere eşitler.

Beyaz alan karakterleri ve karşılaştırma işleçleri yasaklanmıştır. Bir özellik adında gömülü boş değere izin verilir, ancak önerilmez. Gömülü boş değerler kullanırsanız, bu durum, değişken uzunluklu dizgileri belirtmek için MQCHARV yapısıyla birlikte kullanıldığında MQVS_NULL_TERMINATED değişmezinin kullanılmasını önler.

Uygulamalar özellik adlarına dayalı iletileri seçebildiğinden ve seçiciye ilişkin ad ile seçicinin karakter kümesi arasındaki dönüştürmenin beklenmeyen bir şekilde başarısız olmasına neden olabileceğinden, özellik adlarını basit tutun.

WebSphere MQ özellik adları, özelliklerin mantıksal gruplaması için U+002E karakterini (.) kullanır. Bu işlem, ad alanını özellikler için böler. Küçük ya da büyük harflerin herhangi bir karışımında aşağıdaki önekleri içeren özellikler, ürün tarafından kullanılmak üzere ayrılır:

- `mcd`
- `jms`
- `usr`
- `mq`
- `sib`
- `wmq`
- `Root`
- `Body`
- `Properties`

Ad çakışmalarını önlemek için iyi bir yol, tüm uygulamaların ileti özelliklerinin İnternet etki alanı adlarıyla önlerini önlediğinden emin olun. Örneğin, "`ourcompany.com`" etki alanı adını kullanarak bir uygulama geliştiriyorsanız, "`com.ourcompany`" önekine sahip tüm özellikleri adlayabildiniz. Bu adlandırma kuralı, özelliklerin kolay seçilmesine de olanak sağlar; örneğin, bir uygulama "`com.ourcompany.%`" başlangıç olarak başlayan tüm ileti özelliklerini sorgulayabilir.

Özellik adlarının kullanımına ilişkin ek bilgi için Özellik adı kısıtlamaları başlıklı konuya bakın.

Özellik adı kısıtlamaları
Bir özelliği adladığınızda, bazı kuralları izlemeniz gerekir.

Özellik adları için aşağıdaki kısıtlamalar geçerlidir:

1. Bir özellik aşağıdaki dizgilerle başlamamalıdır:
   - "JMS"-JMS için WebSphere MQ sınıfları tarafından kullanılmak üzere ayrılmıştır.
   - "usr.JMS"-geçerli değil.

   JMS özellikleri eşanlamlılarını sağlayan aşağıdaki özelliklerden oluşan özel durumlar şunlardır:

Özellik	**Eşanlamlı**
JMSCorrelationID	Kök .MQMD.CorrelId ya da jms.Cid
JMSDeliveryMode	Kök .MQMD.Persistence ya da jms.Dlv
JMSHedef	jms.Dst
JMSSüresi	Kök .MQMD.Expiry ya da jms.Exp

Özellik	Eşanlamlı
JMSMessageID	Kök .MQMD.MsgId
JMSönceliği	Kök .MQMD.Priority ya da jms.Pri
JMSRedird	Kök .MQMD.BackoutCount
JMSReplyTo (URI olarak kodlanmış bir dizgi)	Kök .MQMD.ReplyToQ ya da Kök .MQMD.ReplyToQMgr ya da jms.Rto
JMSTimestamp	Kök .MQMD.PutDate ya da Root .MQMD.PutTime ya da jms.Tms
JMSType	mcd.Type ya da mcd.Set ya da mcd.Fmt
JMSXAppID	Kök .MQMD.PutApplName
JMSXDeliveryCount	Kök .MQMD.BackoutCount
JMSXGroupID	Kök .MQMD.GroupId ya da jms.Gid
JMSXGroupSeq	Kök .MQMD.MsgSeqNumber ya da jms.Seq
JMSXUserID	Kök .MQMD.UserIdentifier

Bu eşanlamlılar, bir MQI uygulamasının JMS özelliklerine benzer bir şekilde WebSphere MQ sınıflarına JMS istemci uygulaması için erişmesini sağlar. Bu özelliklerin yalnızca JMSCorrelationID, JMSReplyTo, JMSType, JMSXGroupIDve JMSXGroupSeq MQI kullanılarak ayarlanabileceği.

JMS için WebSphere MQ sınıflarında bulunan JMS_IBM_ * özelliklerinin MQI kullanılarak kullanılamadığına dikkat edin. JMS_IBM_ * properties başvurusına, MQI uygulamaları tarafından başka yollarla erişilebilir.

2. Alt ya da büyük harflerin herhangi bir karışımında "NULL", "TRUE", "FALSE", "NOT", "AND", "OR", "BETWELE", "LIKE", "IN", "IS" ve "ESCAPE" gibi bir özellik çağrılmamalı. Bu , seçim dizgilerinde kullanılan SQL anahtar sözcüklerinin adlarıdır.
3. "mq " bir başlangıç özelliği adı "mq_usr" küçük harf ya da büyük harf karışımında yalnızca tek bir "." karakterini içerebilir. karakter (U+002E). Çoklu "." bu önekleri içeren özelliklerde karakterlerin kullanılmasına izin verilmez.
4. İki "." Karakterler arasında başka karakterler de içermeli; sıradüzende boş bir noktana sahip olamazsınız. Benzer şekilde bir özellik adı "." ile bitemez. karakterini kullanın.
5. bir uygulama "a.b" özelliğini ve daha sonra "a.b.c" özelliğini ayarlarsa, "b" sıradüzeninde bir değer mi yoksa başka bir mantıksal gruplama mı () içerip içermediği belirsiz olur. Bu tür bir sıradüzen "karışık içerik" ve bu desteklenmez. Karma içeriğe neden olan bir özelliği ayarlamasına izin verilmez.

Bu kısıtlamaları doğrulama mekanizması tarafından aşağıdaki gibi zorlanır:

- Property names are validated when setting a property using the MQSETMP-İleti özelliğini ayarla call, if validation was requested when the message handle was created. If an attempt to validate a property is undertaken and fails due to an error in the specification of the property name, the completion code is MQCC_FAILED with reason:
  - 1-4 nedenleri için MQRC_XX_ENCODE_CASE_ONE Property_name_error.
  - MQRC_MIXED_CONTENT_NOT_ALLOWED, neden 5.
- Doğrudan MQRFH2 öğeleri olarak belirtilen özelliklerin adları, MQPUT çağrısıyla doğrulanmasını garantilmiyor.

Özellik olarak ileti tanımlayıcı alanları
Çoğu ileti tanımlayıcı alanı özellik olarak değerlendirilebilir. Özellik adı, ileti tanımlayıcısı alanının adına bir önek eklenerek oluşturulur.

Bir MQI uygulaması, bir ileti tanımlayıcı alanında bulunan bir ileti özelliğini tanımlamak isterse (örneğin, bir seçici dizgisinde ya da ileti özelliği API ' lerini kullanarak), aşağıdaki sözdizimini kullanın:

Özellik adı	İleti tanımlayıcı alanı
Root.MQMD. < Alan>	< Alan>

C dili bildirimindeki MQMD yapı alanları için aynı büyük-küçük harf durumu ile `<Field>` değerini belirtin. Örneğin, `Root.MQMD.AccountingToken` özellik adı, ileti tanımlayıcısının `AccountingToken` alanına erişir.

İleti tanımlayıcısının `StrucId` ve `Version` alanlarında gösterilen sözdizimi kullanılarak erişilebilir değil.

İleti tanımlayıcı alanları, diğer özellikler için hiçbir zaman bir MQRFH2 üstbilgisinde gösterilmez.

İleti verileri kuyruk yöneticisi tarafından onurlandırılan bir MQMDE ile başlıyorsa, açıklanan `Root.MQMD.<Field>` gösterimi kullanılarak MQMDE alanlarına erişilebilir. Bu durumda, MQMDE alanları, bir özellikler perspektifinden MQMD ' nin mantıksal olarak kabul edilir. MQMDE ' ye Genel Bakışiçindeki "MQPUT ve MQPUT1 çağrılarında belirtilen MQMDE" kısmına bakın.

Özellik veri tipleri ve değerleri

Bir özellik, boole, bayt dizilimi, karakter dizilimi ya da kayan noktalı ya da tamsayı olabilir. Bağlam, bağlam tarafından aksi belirtilmediği sürece, veri tipi aralığında geçerli herhangi bir değeri saklayabilir.

Bir özellik değerinin veri tipi, aşağıdaki değerlerden biri olmalıdır:

- MQBOOL
- MQBYTE []
- MQCHAR []
- MQFLOAT32
- MQFLOAT64
- MQINT8
- MQINT16
- MQINT32
- MQINT64

Bir özellik var, ancak tanımlı bir değeri yok; boş değerli bir özeldir. Boş değerli bir özellik, bir byte özelliğinden (MQBYTE []) ya da karakter dizgisi özelliğinden (MQCHAR []) farklı, ancak değeri sıfır olan bir değeri olan boş bir değer içeriyor.

Bayt dizilimi, JMS ya da XMSiçinde geçerli bir özellik veri tipi değil. < usr> klasöründe byte dizgisi özelliklerini kullanmamanız önerilir.

Kuyruklardan iletilerin seçilmesi

Bir MQGET çağrısındaki MsgId ve CorrelId alanlarını kullanarak ya da bir MQOPED ya da MQSUB çağrısında SelectionString kullanarak, kuyruklardan ileti seçebilirsiniz.

Seçiciler

İleti seçici, bir uygulamanın ilgisini yalnızca, seçim dizgisinin temsil ettiği SQL (Yapılandırılmış Sorgu Dili) sorgusunu karşılayan özelliklere sahip iletilere kaydettirmek için kullanılan değişken uzunluklu bir dizilimdir.

MQSUB ve MQOPEN işlev çağrılarını kullanarak seçim

MQSUB ve MQOP çağrılarını kullanarak seçim yapmak için MQCHARV tipinde bir yapı olan *`SelectionString`*' ı kullanıyorsunuz.

The *`SelectionString`* structure is used to pass a variable-length selection string to the queue manager.

Seçici dizgisiyle ilişkilendirilmiş CCSID, MQCHARV yapısının VSCCSID alanı aracılığıyla ayarlanır. Kullanılan değer, seçici dizgileri için desteklenen bir CCSID olmalıdır. Desteklenen kod sayfalarının listesi için Kod sayfası dönüşümü başlıklı konuya bakın.

Specifying a CCSID for which there is no WebSphere MQ supported Unicode conversion, results in an error of MQRC_SOURCE_CCSID_ERROR. Bu hata, seçici kuyruk yöneticisine (MQSUB, MQOPEN ya da MQPUT1 üzerinde) sunulmaya başlansa döndürülür.

VSCCSID alanının varsayılan değeri, seçim dizgisinin CCSID değerinin kuyruk yöneticisi CCSID değerine eşit olduğunu ya da bir istemci üzerinden bağlandıysa istemci CCSID değerinin eşit olduğunu gösterir. MQCCSI_APPL değişmezi, derlemeden önce bir uygulama yeniden tanımlanarak geçersiz kılınabilir.

MQCHARV seçicisi bir NULL dizgisi gösteriyorsa, o ileti tüketicisi için seçim gerçekleşmez ve iletiler, bir seçici kullanılmamış gibi teslim edilir.

Bir seçim dizgisinin uzunluk üst sınırı yalnızca, *VSLengthMQCHARV* alanı tarafından tanımlanabilen bir dizilimle sınırlandırılır.

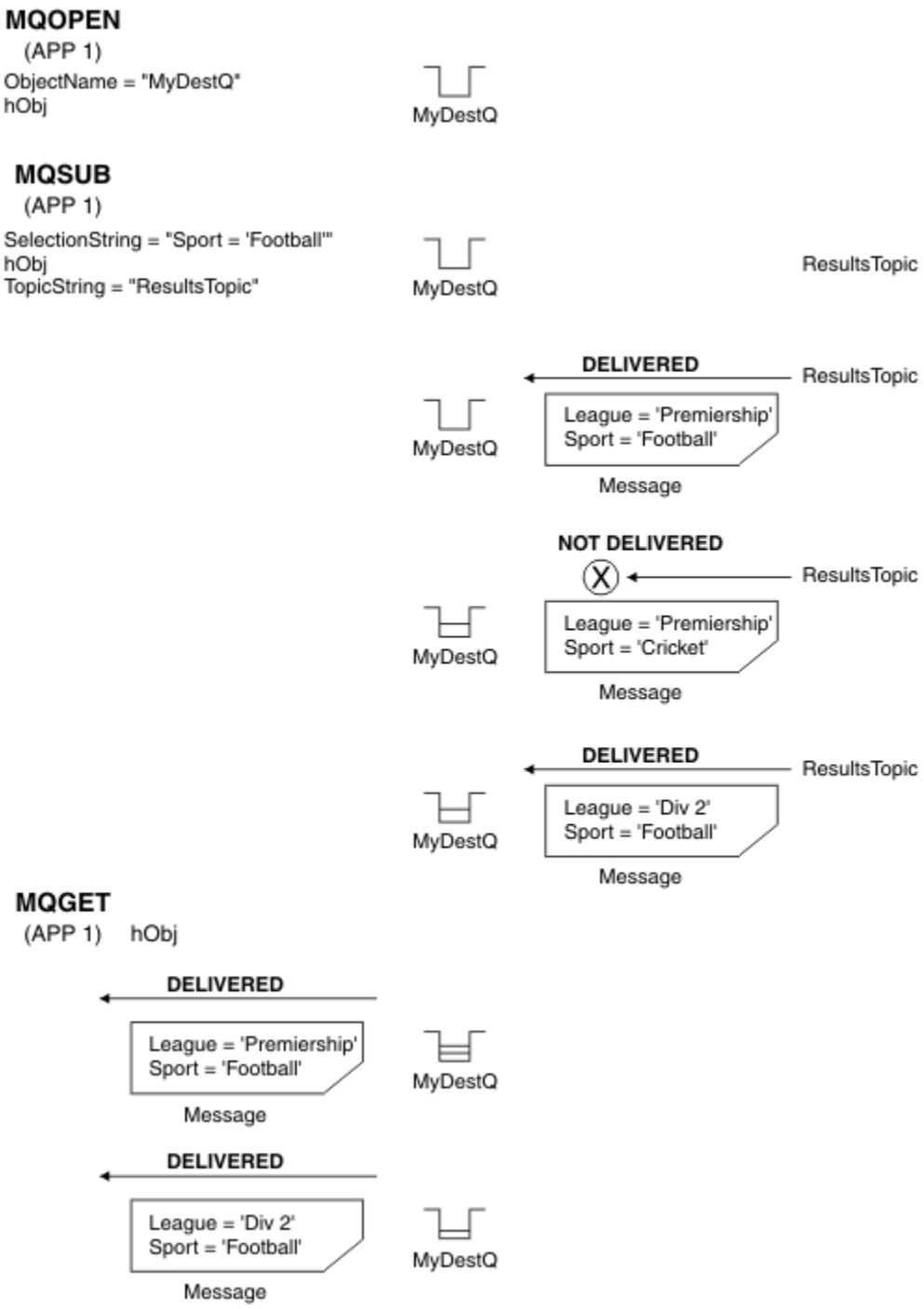
Bir arabellek sağladıysanız ve VSBufSizeiçinde artı bir arabellek uzunluğu varsa, MQSO_RESUME abone olma seçeneğini kullanarak bir MQSUB çağrısından çıkışta SelectionString döndürülür. Bir arabellek sağlamadıysanız, MQCHARV ' ın VSLecgth alanında yalnızca seçim dizgisinin uzunluğu döndürülür. Sağlanan arabellek, alanı döndürmek için gereken alandan küçükse, sağlanan arabellete yalnızca VSBufSize byte değeri döndürülür.

Bir uygulama, ilk olarak kuyruğun tanıtıcısını (MQOPEN için) ya da aboneliği kapatmaksızın (MQSUB için) bir seçim dizgisini değiştiremiyor. Daha sonra, sonraki bir MQOPED ya da MQSUB çağrısında yeni bir seçim dizgisi belirtilebilir.

MQOPEN
Açılan tanıtıcıyı kapatmak için MQCLOSE komutunu kullanın ve daha sonra, sonraki bir MQOPEN çağrısında yeni bir seçim dizgisi belirtin.

MQSUB
Döndürülen abonelik tanıtıcısını (hSub) kapatmak için MQCLOSE komutunu kullanın ve daha sonra, sonraki bir MQSUB çağrısında yeni bir seçim dizgisi belirtin.

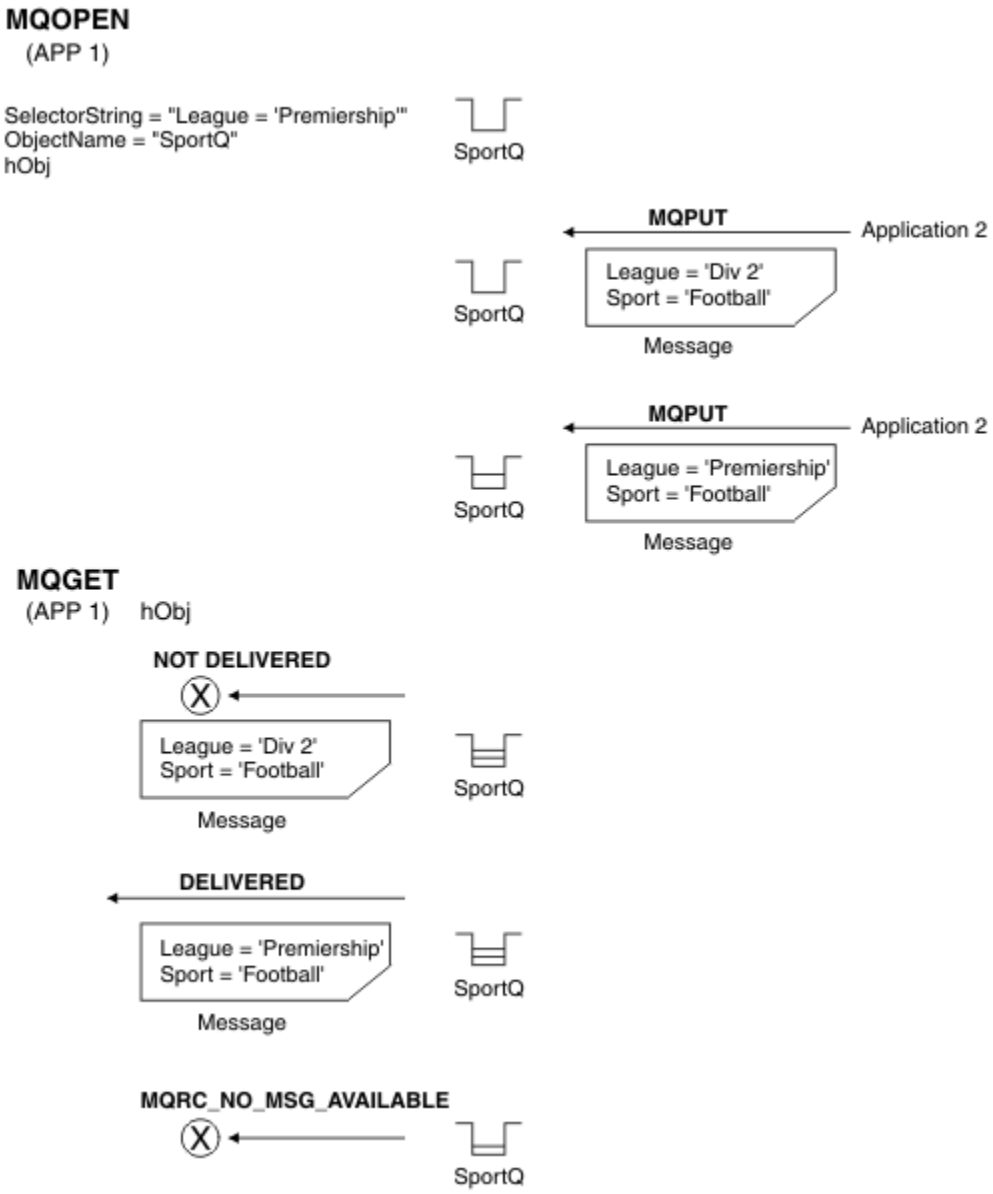
Şekil 3 sayfa 23 , MQSUB çağrısını kullanarak seçim sürecini gösterir.



Şekil 3. MQSUB çağrısını kullanarak seçim

MQSD yapısındaki *SelectionString* alanı kullanılarak MQSUB çağrısına bir seçici aktarılabilir. MQSUB ' da bir seçicide geçirmenin etkisi, yalnızca sağlanan konuya abone olunan ve sağlanan seçim dizgisiyle eşleşen iletilerin hedef kuyrukta kullanılabilir kılındığı iletilerdir.

Şekil 4 sayfa 24 , MQOPEN çağrısını kullanarak seçim işlemini gösterir.



Şekil 4. MQOPER çağrısını kullanarak seçim

MQOD yapısındaki *SelectionString* alanını kullanarak, MQOPEN çağrısına bir seçici aktarılabilir. MQOPER çağrısında bir seçicide geçirmenin etkisi, yalnızca açılan kuyrukta bulunan ve bir seçiciyle eşleşen iletilerin ileti tüketicisine teslim edilmeleridir.

Bir uygulamanın yalnızca bir seçiciyle eşleşen bir kuyruktaki iletileri almayı seçebileceği noktadan noktaya iletişim kutusu için, MQOPEN çağrısında seçici için ana kullanım kullanılır. Önceki örnek, iki iletinin MQOPER tarafından açılan bir kuyruğa yerleştirildiği, ancak bir seçiciyle eşleşen tek bir senaryonun yalnızca biri tarafından alındığı basit bir senaryoya yol açar.

İzleyen MQGET çağrılarının MQRC_NO_MSG_AVAILEABLE ile, belirtilen seçiciyle eşleşen kuyrukta başka ileti olmadığını belirtmesine dikkat edin.

Seçim davranışı
IBM WebSphere MQ seçim davranışına genel bakış.

MQMD yapıldıysa, bir MQMDE yapısındaki alanlar, MQMD ' ye karşılık gelen ileti tanımlayıcısı özelliklerine ilişkin ileti özellikleri olarak kabul edilir:

- MQFMT_MD_EXTENSION biçimi var

- Hemen ardından geçerli bir MQMDE yapısı var
- Sürüm bir ya da yalnızca iki alan için varsayılan sürüm olan iki alan içerir

Herhangi bir ileti özelliği gerçekleşmeden önce seçim dizgisinin TRUE ya da FALSE olarak çözülmesi mümkündür., Örneğin, seçim dizgisi "TRUE <>FALSE"olarak ayarlandıysa, bu durum söz sahibi olabilir. Bu tür erken değerlendirmenin, yalnızca seçim dizgisinde ileti özelliği başvuruları olmadığında gerçekleşeceği garanti edilir.

Bir seçim dizgisi, herhangi bir ileti özelliği dikkate alınmadan önce TRUE (Doğru) değerine çözülürse, tüketici tarafından abone olunan konuya yayınlanan tüm iletiler teslim edilir. Bir seçim dizgisi, herhangi bir ileti özelliği dikkate alınmadan önce FALSE değerine çözülürse, seçiciyi sunan işlev çağrısında MQRC_SELECTOR_ALWAY_FALSE ve tamamlanma kodu MQCC_FAILED bir neden kodu döndürülür.

Bir ileti ileti özelliği içermiyorsa (üstbilgi özellikleri dışında), seçim için uygun olabilir. Bir seçim dizgisi var olmayan bir ileti özelliğine başvuruyorsa, bu özelliğin boş değer (NULL) ya da 'Unknown ' (Bilinmiyor) değerine sahip olduğu varsayılır.

Örneğin, bir ileti yine de `'Color IS NULL'` gibi bir seçim dizgisini karşılayabilir; burada `'Color'` , iletide bir ileti özelliği olarak var olmaz.

Seçim yalnızca, genişletilmiş bir ileti seçimi sağlayıcısı yoksa, iletinin kendisi değil, yalnızca bir iletiyle ilişkili özelliklerde gerçekleştirilebilir. Yalnızca genişletilmiş bir ileti seçimi sağlayıcısı varsa, ileti bilgi yükinde seçim gerçekleştirilebilir.

Her ileti özelliğinin, kendisiyle ilişkilendirilmiş bir tipi vardır. Bir seçim gerçekleştirdiğinizde, ileti özelliklerini test etmek için ifadelerde kullanılan değerlerin doğru tipte olduğundan emin olmanız gerekir. Tip uyuşmazlığı ortaya çıkarsa, söz konusu ifade FALSE olarak çözülüyor.

Seçim dizgisinin ve ileti özelliklerinin uyumlu tipler kullandığından emin olmak sizin sorumluluğunuzda.

Seçim ölçütleri, etkin olmayan dayanıklı aboneler adına uygulanmaya devam eder, böylece yalnızca başlangıçta sağlanan seçim dizgisiyle eşleşen iletiler saklanır.

Kalıcı abonelik alter (MQSO_ALTER) ile sürdürüldüğünde seçim dizgileri değiştirilemez. Bir kalıcı abone etkinliği devam ettiğinde farklı bir seçim dizgisi sunulursa, uygulamaya MQRC_SELECTOR_NOT_ALTERABLE değeri döndürülür.

Bir kuyruktan seçim ölçütlerine uygun bir ileti yoksa, uygulamalar MQRC_NO_MSG_AVAIABLE dönüş kodunu alır.

Bir uygulama, özellik değerlerini içeren bir seçim dizgisi belirttiyse, yalnızca eşleşen özellikleri içeren iletiler seçime uygun olur. Örneğin, bir abone "a = 3" seçim dizgisini, hiçbir özellik içermeyen bir ileti yayınlanıyor ya da 'a' var olmayan ya da 3 'e eşit olmayan özellikler içeriyor. Abone, hedef kuyruğuna bu iletiyi almaz.

İleti alışverişi başarımı

Selecting messages from a queue requires IBM WebSphere MQ to sequentially inspect each message on the queue. İletiler, seçim ölçütleriyle eşleşen bir ileti bulununcaya ya da incelenecek başka ileti bulununcaya kadar incelenir. Bu nedenle, ileti seçimi derin kuyruklarda kullanılırsa, ileti alışverişi başarımı zarar görür.

To optimize message selection on deep queues when selection is based on JMSCorrelationID or JMSMessageID, use a selection string of the form `JMSCorrelationID = ...` or `JMSMessageID = ...` and reference only one property.

Bu yöntem, JMSCorrelationID ile ilgili seçim performansında önemli bir gelişme sunar ve JMSMessageIDiçin marjinal bir başarım artışı sağlar.

Karmaşık seçicilerin kullanılması

Seçiciler birçok bileşen içerebilir; örneğin:

a and b or c and d or e and `f` or g and h or `i` and `j` ... ya da y ve z

Bu tür karmaşık seçicilerin kullanımı ciddi performans etkileri ve aşırı kaynak gereksinimlerine sahip olabilir. Bu nedenle, IBM WebSphere MQ sistemi, sistem kaynağı eksiklik sonuçlarıyla sonuçlanabilen aşırı karmaşık seçicileri işleyemeyerek koruyacaktır. Koruma, bazı platformlarda yapılan yaklaşık 100 sınamadan sonra ortaya çıkabilir, böylece bu bileşen sayısına yaklaşan seçiciler arızalar görebilir. Birçok bileşene sahip seçicilerin kullanılması, koruma sınırlarına ulaşılmamasını sağlamak için uygun altyapılarda ayrıntılı bir şekilde denenir ve sınanmış olur.

Seçicilerin performansı ve karmaşıklığı, bileşenleri birleştirmek için ek parantez kullanarak basitleştirilerek geliştirilebilir. Örneğin:

(a ve b ya da c ve d) ya da (e ve f ya da g ve h) ya da (i ve j) ...

İlgili kavramlar

İleti seçici sözdizimi

WebSphere MQ ileti seçici, SQL92 koşullu ifade sözdiziminin bir alt kümesini temel alan sözdizimine sahip bir dizgidir.

İletinin içeriğini seçme

Bir ileti bilgi yükü içeriğine (içerik süzgeci olarak da bilinir) dayalı olarak abone olmak mümkündür; ancak, bu tür bir aboneliğe teslim edilmesi gereken iletilerin doğrudan WebSphere MQtarafından gerçekleştirilememesi gerekir; örneğin, iletilerin işlenmesi için genişletilmiş bir ileti seçimi sağlayıcısı (örneğin, IBM Integration Bus) gerekir.

İleti seçici sözdizimi

WebSphere MQ ileti seçici, SQL92 koşullu ifade sözdiziminin bir alt kümesini temel alan sözdizimine sahip bir dizgidir.

Bir ileti seçicinin değerlendirdiği sıra, öncelik düzeyi içinde soldan sağa doğru olur. Bu siparişi değiştirmek için ayraçları kullanabilirsiniz. Önceden tanımlı seçici hazır bilgileri ve işleç adları burada büyük harfle yazılır; ancak, büyük/küçük harf duyarlı değildir.

WebSphere MQ , sunulma sırasında bir ileti seçicisinin sözdizimsel doğruluğu doğrulatır. If the syntax of the selection string is incorrect or a property name is not valid, and an extended message selection provider is not available, `MQRC_SELECTION_NOT_AVAILABLE` is returned to the application. Seçim dizgisinin sözdizimi yanlışsa ya da abonelik sürdürüldüğünde bir özellik adı geçerliyse, uygulamaya bir `MQRC_SELECTOR_SYNTAX_ERROR` döndürülür. Özellik ayarlanırken özellik adı geçerlilik denetimi geçersiz kılındıysa (`MQCMHO_VALIDATE`yerine `MQCMHO_NONE` belirlenerek) ve bir uygulama daha sonra geçersiz özellik adına sahip bir ileti yerleştirdiyse, bu ileti hiçbir zaman seçilmez.

Bir seçici şunları içerebilir:

- Hazır bilgiler:
  - Dizgi hazır bilgileri tek tırnak işareti içine alınır. Birbirini izleyen iki tek tırnak işareti tek tırnak işaretini temsil eder. 'Literal' ve 'literal' (hazır bilgi) örnekleri verilebilir. Java dizgi hazır bilgileri gibi, bunlar Unicode karakter kodlamasını kullanır. Bir dizgi hazır bilgisini kapatmak için çift tırnak işareti kullanamazsınız. Tek tırnak işaretleri arasında herhangi bir bayt dizisi kullanılabilir.
  - Bir bayt dizilimi, çift tırnak işareti içine alınmış ve önünde `0x`öneki bulunan bir ya da daha çok onaltılı karakter çiftidir. Örnekler: `"0x2F1C"` ya da `"0XD43A"`. Bayt dizilimi uzunluğunun en az bir bayt olması gerekir. Bir seçici byte dizgisi `MQTYPE_BYTE_STRING`tipindeki bir ileti özelliğiyle eşleştirilirse, baştaki ya da sondaki sıfırda özel bir işlem yapılmamaktadır. Byte 'lar başka bir karakter olarak işlem görür. Endianness de dikkate alınmıyor. Seçici ve özellik byte dizgilerinin uzunluğu eşit olmalıdır ve byte sırası aynı olmalıdır.

    Bayt dizilimi seçimlerine ilişkin örnekler (*myBytes* = `0AFC23`) aşağıdaki gibi olur:

    - `"myBytes = "0x0AFC23"" = TRUE`

    Aşağıdaki dizgi seçimleri eşleşmiyor:

    - `"myBytes = "0xAFC23"" = MQRC_SELECTOR_SYNTAX_ERROR` (çünkü byte sayısı birden çok iki byte değil)
    - `"myBytes = "0x0AFC2300"" = FALSE` (karşılaştırmada sondaki sıfırın önemli olduğu için)

- "myBytes = "0x000AFC23"" = FALSE (karşılaştırmada sıfır değeri önemli olduğu için)
- "myBytes = "0x23FC0A"" = FALSE (endianness dikkate alınmadığı için)

– Onaltılı sayılar sıfır ile başlar ve ardından büyük ya da küçük harfli xolur. Hazır bilginin geri kalan kısmı bir ya da daha çok geçerli onaltılı karakter içeriyor. Örnekler: 0xA, 0xAF, 0X2020.

– 0-7 aralığındaki bir ya da daha çok basamak izleyen bir ya da daha çok basamak, her zaman sekizli bir sayının başlangıcı olarak yorumlanır. Sıfır önekli bir ondalık sayıyı bu gibi gösteremezsiniz; örneğin, 09 , 9 geçerli bir sekizli sayı olmadığı için bir sözdizimi hatası döndürür. Sekizli sayı örnekleri: 0177, 0713.

– An exact numeric literal is a numeric value without a decimal point, such as 57, -957, and +62. Tam sayısal hazır bilgi, sondaki büyük ya da küçük harflere sahip olabilir; bu, sayının nasıl depolanır ya da yorumlanabileceği etkilemez. WebSphere MQ supports exact numerals in the range -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.

– An approximate numeric literal is a numeric value in scientific notation, such as 7E3 or -57.9E2, or a numeric value with a decimal, such as 7., -95.7, or +6.2. WebSphere MQ supports numbers in the range -1.797693134862315E+308 to 1.797693134862315E+308.

  Bu değer, isteğe bağlı bir işaret karakterinin (+ ya da -) izlenmesi gerekir. Değer, tamsayı ya da kesir olmalıdır. Çok önemli bir kesirli kısım ve başında gelen bir basamak yok.

  Büyük ya da küçük harfli E , isteğe bağlı bir üslubunun başlangıcını belirtir. Üstel bir ondalık radx ve üstel sayı parçası, isteğe bağlı bir işaret karakteriyle önek olarak öner olabilir.

  Yaklaşık sayısal hazır bilgiler F ya da D karakteri tarafından sonlandırılabilir (büyük/küçük harfe duyarlı değildir). Bu sözdizimi, tek ya da çift duyarlıklı sayıların etiketlendirilmesinin dil arası yöntemini desteklemesi için vardır. Bu karakterler isteğe bağlıdır ve yaklaşık sayısal hazır bilginin nasıl saklanmayacağını ya da işleneceğini etkilemez. Bu sayılar her zaman çift duyarlıklı olarak depolanır ve işlenir.

– Boole hazır bilgileri TRUE ve FALSE.

Not: Non-finite IEEE-754 representations such as NaN, +Infinity, -Infinity are not supported in selection strings. Bu nedenle, bu değerlerin bir ifadede işlenenler olarak kullanılması mümkün değildir. Negatif sıfır, matematiksel işlemler için pozitif sıfır olarak ele alınır.

- Tanıtıcılar:

  Tanıtıcı, geçerli bir tanıtıcı başlangıç karakteriyle başlaması gereken, sıfır ya da daha fazla geçerli tanıtıcı parça karakteriyle başlaması gereken değişken uzunluklu bir karakter dizidir. Tanıtıcı adlarına ilişkin kurallar, ileti özelliği adları için aynıdır, ek bilgi için bkz. “Özellik adları” sayfa 18 ve “Özellik adı kısıtlamaları” sayfa 19 .

  Not: Yalnızca genişletilmiş bir ileti seçimi sağlayıcısı varsa, ileti bilgi yükinde seçim gerçekleştirilebilir.

  Tanıtıcılar, üstbilgi alanı başvuruları ya da özellik başvurularıdır. Bir ileti seçicindeki bir özellik değerinin tipi, mümkün olduğu yerlerde sayısal promosyon gerçekleştirilmesine rağmen, özelliği ayarlamak için kullanılan tipe karşılık gelmelidir. Tip uyuşmazlığı ortaya çıkarsa, ifadenin sonucu FALSEolur. İletide var olmayan bir özelliğe başvurulsa, değeri NULLolur.

  Bir ileti seçici ifadesinde bir özellik kullanıldığında, özellikler için alma yöntemleri için geçerli olan tip dönüştürmeleri geçerli olmaz. Örneğin, bir özellik dizgi değeri olarak ayarlandıysa ve bunu sayısal değer olarak sorgulamak için bir seçici kullanırsanız, ifade FALSEdeğerini döndürür.

  Özellik adlarıyla ya da MQMD alan adlarıyla eşlenen JMS alanı ve özellik adları, bir seçim dizgisinde de geçerli tanıtıcılardır. WebSphere MQ , tanınan JMS alanını ve özellik adlarını ileti özelliği değerleriyle eşler. Ek bilgi için “JMS ' de ileti seçicileri” sayfa 775 başlıklı konuya bakın. As an example, the selection string "JMSPriority >=" selects on the Pri property found in the jms folder of the current message.

- Taşma/alt akış:

  Hem ondalık hem de yaklaşık sayısal sayılar için, aşağıdakiler tanımsız olur:

  – Tanımlı aralık dışında bir sayı belirtme

  – Taşmaya ya da alt akışa neden olacak bir aritmetik ifade belirtilmesi

Bu koşullar için denetim gerçekleştirilmez.

- Beyaz alan:

  Boşluk, form besleme, yeni satır, satır başı, yatay sekme ya da dikey sekme olarak tanımlanır. Aşağıdaki Unicode karakterleri beyaz alan olarak tanınır:

  - `\u0009 to \u000D`
  - `\u0020`
  - `\u001C`
  - `\u001D`
  - `\u001E`
  - `\u001F`
  - `\u1680`
  - `\u180E`
  - `\u2000 - \u200A`
  - `\u2028`
  - `\u2029`
  - `\u202F`
  - `\u205F`
  - `\u3000`
- İfadeler:
  - Seçici, koşullu bir ifadedir. Gerçek eşleşmeleri değerlendiren bir seçici; yanlış ya da bilinmeyen olarak değerlendirilen bir seçici eşleşmez.
  - Aritmetik ifadeler, kendilerinden oluşur, aritmetik işlemler, tanıtıcılar (tanıtıcı değeri sayısal hazır bilgi olarak kabul edilir) ve sayısal hazır bilgiler.
  - Koşullu ifadeler kendilerinden, karşılaştırma işlemlerinden ve mantıksal işlemlerden oluşur.
- İfadelerin değerlendirilen sıralamayı ayarlamak için standart destek pazarlama () desteklenir.
- Mantıksal işleçler öncelik sırasına göre: `NOT`, `AND`, `OR`.
- Karşılaştırma işleçleri: =, >, >=, <, <=, <> (eşit değil).
  - İki baytlık dizgi yalnızca dizgiler aynı uzunlukdaysa ve byte sırası eşitse eşittir.
  - Yalnızca aynı tipteki değerler karşılaştırılabilir. Bir kural dışı durum, tam sayısal değerleri ve yaklaşık sayısal değerleri karşılaştırmak için geçerlidir (gereken tür dönüştürme, Java sayısal promosyonunun kuralları tarafından tanımlanır). Farklı tipleri karşılaştırma girişiminde bulunulursa, seçici her zaman false olur.
  - Dizgi ve boole karşılaştırması = ve <>ile sınırlandırılmıştır. İki dizgi, yalnızca aynı karakter dizisini içerirse eşittir.
- Aritmetik işleçler öncelik sırasına göre:
  - +, - birli.
  - * çarpma ve / bölümü.
  - + ekleme ve - çıkarma.
  - Boş değer (NULL) olan aritmetik işlemler desteklenmez. Bunlar denenirse, tam seçici her zaman false olur.
  - Aritmetik işlemler, Java sayısal promosyonu kullanmalıdır.
- arithmetic-expr1 [`NOT`] `BETWEEN` arithmetic-expr2 ve arithmetic-expr3 karşılaştırma işleci:
  - `Age BETWEEN 15 and 19`, `age >= 15 AND age <= 19`ile eşdeğerdir.
  - `Age NOT BETWEEN 15 and 19`, `age < 15 OR age > 19`ile eşdeğerdir.

  - Bir BETWEEN işleminin ifadelerinden herhangi biri NULL (boş) ise, işlemin değeri false olur. Bir NOT BETWEEN işleminin ifadelerinden herhangi biri NULLolursa, işlemin değeri doğrudur.
- tanıtıcı [NOT] IN (string-literal1, string-literal2,...) karşılaştırma işlecinde bir Dize ya da NULL değeri var.
  - Country IN ('UK', 'US', 'France') , 'UK' için true, 'Peru'için false. Bu, (Country = 'UK') OR (Country = 'US') OR (Country = 'France')ifadesiyle eşdeğerdir.
  - Country NOT IN ('UK', 'US', 'France') is false for 'UK' and true for 'Peru'. Bu, NOT ((Country = 'UK') OR (Country = 'US') OR (Country = 'France'))ifadesiyle eşdeğerdir.
  - Bir IN ya da NOT IN işleminin tanıtıcısı NULL ise, işlemin değeri bilinmez.
- identifier [NOT] LIKE *pattern-value* [ESCAPE *escape-character*] karşılaştırma işleci, burada identifier bir dizgi değerine sahiptir. *pattern-value* , bir dizgi hazır bilgisidir; burada _ herhangi bir tek karakter için, % ise herhangi bir karakter dizisi (boş sıra dahil) için durmaktadır. Diğer tüm karakterler kendileri için geçerli. The optional *çıkış karakteri* is a single character string literal that is used to escape the special meaning of the _ and % in *örüntü-değer*. LIKE işlecinin yalnızca iki dizgi değerini karşılaştırmak için kullanılması gerekir.
  - phone LIKE '12%3' , 123 ve 12993 için geçerlidir ve 1234 için false.
  - word LIKE 'l_se' , kaybedilen ve gevşek için false (yanlış) için geçerlidir.
  - underscored LIKE '_%' ESCAPE '\' , _foo için true, bariçin false.
  - phone NOT LIKE '12%3' is false for 123 and 12993 and true for 1234.
  - Bir LIKE ya da NOT LIKE işleminin tanıtıcısı NULLise, işlemin değeri bilinmez.

  Not: İki dizgi değerini karşılaştırmak için LIKE işleci kullanılmalıdır. Root.MQMD.CorrelId değeri, bir karakter dizgisi değil, 24 byte 'lık bir bayt dizisidir. The selector string Root.MQMD.CorrelId LIKE 'ABC%' is accepted by the parser as syntactically valid, but it is evaluated to false. Bir bayt dizisini karakter dizgisiyle karşılaştırırken, LIKE kullanılamaz.
- Bir NULL üstbilgisi alan değeri ya da eksik bir özellik değeri içinidentifier IS NULL karşılaştırma işleci testleri.
- identifier IS NOT NULL comparison operator tests for the existence of a non-null header field value or a property value.
- Boş değerler

  NULL değerlerini içeren seçici ifadelerinin değerlendirilmesi, özet olarak SQL 92 NULL anlambilimi ile tanımlanır:

  - SQL, bir NULL değerini bilinmeyen olarak değerlendirir.
  - Bilinmeyen bir değere sahip karşılaştırma ya da aritmetik, her zaman bilinmeyen bir değer verir.
  - IS NULL ve IS NOT NULL işleçleri, bilinmeyen bir değeri TRUE ve FALSE değerlerine dönüştürür.

  Boole işleçleri üç değerli mantığı kullanır (T=TRUE, F=FALSE, U=UNKNOWN)

Çizelge 1. Mantık A AND Bolduğunda Boole işleci sonucu

Operatör A	**Operatör B**	**Sonuç (A VE B)**
T	F	F
T	U	U
T	T	T
F	T	F
F	U	F
F	F	F
U	T	U

Çizelge 1. Mantık A AND Bolduğunda Boole işleci sonucu (devamı var)		
Operatör A	**Operatör B**	**Sonuç (A VE B)**
U	U	U
U	F	F

Çizelge 2. Mantık A OR Bolduğunda Boole işleci sonucu		
Operatör A	**Operatör B**	**Sonuç (A OR B)**
T	F	T
T	U	T
T	T	T
F	T	T
F	U	U
F	F	F
U	T	T
U	U	U
U	F	U

Çizelge 3. Mantık NOT Aolduğunda Boole işleci sonucu	
Operatör A	**Sonuç (NOT A)**
T	F
F	T
U	U

Aşağıdaki ileti seçici, ileti tipi araba, mavi renk ve ağırlığı 2500 lbs 'den büyük olan iletileri seçer:

```
"JMSType = 'car' AND color = 'blue' AND weight > 2500"
```

SQL, sabit ondalık karşılaştırma ve aritmetik desteklerini desteklese de, ileti seçicileri desteklemez. Bu nedenle, tam sayısal hazır bilgiler ondalıklı olmayan bu sayılarla sınırlandırılmıştır. Ayrıca, yaklaşık bir sayısal değer için alternatif bir temsil olarak bir ondalık ile sayısal değerler olmasının nedeni de bu.

SQL açıklamaları desteklenmez.

İlgili kavramlar

Seçim davranışı
IBM WebSphere MQ seçim davranışına genel bakış.

İletinin içeriğini seçme
Bir ileti bilgi yükü içeriğine (içerik süzgeci olarak da bilinir) dayalı olarak abone olmak mümkündür; ancak, bu tür bir aboneliğe teslim edilmesi gereken iletilerin doğrudan WebSphere MQtarafından gerçekleştirilememesi gerekir; örneğin, iletilerin işlenmesi için genişletilmiş bir ileti seçimi sağlayıcısı (örneğin, IBM Integration Bus) gerekir.

“İleti Özellikleri” sayfa 17
Bir uygulamanın işlenecek iletileri seçmesine izin vermek ya da MQMD ya da MQRFH2 üstbilgilerine erişmeden bir iletiyle ilgili bilgileri almak için ileti özelliklerini kullanın. Bunlar, WebSphere MQ ile JMS uygulamaları arasında iletişimi de kolaylaştırır.

İlgili başvurular

MsgHandle

MQBUFMH-Arabelleği ileti tutamaçına dönüştür

Seçim dizgisi kuralları ve kısıtlamaları

Seçim dizgilerinin, seçicileri kullanırken olası sorunları önlemek için nasıl yorumlanır ve karakter kısıtlamaları olduğunu öğrenmek için kendinizi bu kurallara uygun bir şekilde öğrenin.

- Eşdeğerlik, tek bir eşittir karakteri kullanılarak sınanmış; örneğin, `a = b` doğru, `a == b` yanlış.
- Birçok programlama dili tarafından 'eşit değil' temsil etmek üzere kullanılan bir işleç `!=`. Bu gösterim, `<>`için geçerli bir eşanlamlı değil; örneğin `a <> b` geçerli değil, `a != b` geçerli değil.
- Tek tırnak işaretleri yalnızca ' (U+0027) karakteri kullanılır. Benzer şekilde, yalnızca bayt dizilimlerini kapamak için kullanıldığında geçerli olmak üzere çift tırnak imi " (U+0022) karakteri.
- &, &&, | ve || simgeleri mantıksal bağlaç/çıkış için eşanlamlılar değildir; örneğin, `a && b` , `a AND` bolarak belirtilmelidir.
- The wildcard characters `*` and `?` are not synonyms for `%` and `_`.
- `20 < b < 30` gibi bileşik ifadeler içeren seçiciler geçerli değil. Ayrıştırıcı, soldan sağa aynı önceliğe sahip işleçleri değerlendirir. Bu nedenle örnek, anlamlı gelmeyen `(20 < b) < 30`olur. Bunun yerine ifade, `(b > 20) AND (b < 30)`olarak yazılmalıdır.
- Byte dizgileri çift tırnak içine alınmalıdır; tek tırnak işaretleri kullanılırsa, bayt dizilimi dizgi hazır bilgisi olarak alınır. `0x` ' in ardından karakter sayısı (karakterlerin gösterdiği sayı değil), iki karakter olmalıdır.
- `IS` anahtar sözcüğü, eşittir karakteriyle eşanlamlı değil. Thus the selection strings `a IS 3` and `b IS 'red'` are not valid. `IS` anahtar sözcüğü yalnızca `IS NULL` ve `IS NOT NULL` vakalarını desteklemek için vardır.

İlgili kavramlar

İleti seçicileri kullanırkenUTF-8 ve Unicode ile ilgili dikkat edilmesi gereken noktalar

İleti seçicileri kullanırkenUTF-8 ve Unicode ile ilgili dikkat edilmesi gereken noktalar

Tek tırnak işareti içine alınmamış, bir seçim dizgisinin ayrılmış anahtar sözcüklerini oluşturan karakterler, Basic Latin Unicode 'da (U+0000 karakterinden U+0007Farasında değişir) girilmelidir. Alfasayısal karakterlerin diğer kod noktası gösterimlerini kullanmak için geçerli değildir. Örneğin, 1 numaralı sayının Unicode 'da U+0031 olarak ifade edilmesi gerekir, Tam Genişlikli Basamak eşdeğer U+FF11 ya da Arapça eşdeğer U+0661' in kullanılması geçerli değildir.

İleti özelliği adları, herhangi bir Unicode karakter dizisi kullanılarak belirlenebilir. Message property names contained within selection strings that are encoded in UTF-8 will be validated even if they contain multi-byte characters. Çok baytlı UTF-8 için geçerlilik denetimi katıdır ve ileti özelliği adları için geçerli UTF-8 sıralarının kullanıldığından emin olmanız gerekir.

Eşitlik için karşılaştırılırken özellik adlarında ya da değerlerde fazladan işlem gerçekleştirilmez. Bu, örneğin, hiçbir kompozisyonun gerçekleşmediği ve bağlarının hiçbir özel anlam ifade etmediğine dair bir anlam ifade ediyor. Örneğin, önceden oluşturulmuş umlaut karakteri U+00FC , U+0075 + U+0308 karakterlerine eşdeğer olarak kabul edilmiyor ve karakter sırası ff değerinin Unicode U+FB00 (LATIN KüçüK BAğLANTı FF) ile eşdeğer olduğu kabul edilmez.

Tek tırnak işaretleri içindeki özellik verileri, herhangi bir bayt dizisi ile gösterilebilir ve geçerliliği denetlenmez.

İlgili kavramlar

Seçim dizgisi kuralları ve kısıtlamaları

Seçim dizgilerinin, seçicileri kullanırken olası sorunları önlemek için nasıl yorumlanır ve karakter kısıtlamaları olduğunu öğrenmek için kendinizi bu kurallara uygun bir şekilde öğrenin.

İletinin içeriğini seçme

Bir ileti bilgi yükü içeriğine (içerik süzgeci olarak da bilinir) dayalı olarak abone olmak mümkündür; ancak, bu tür bir aboneliğe teslim edilmesi gereken iletilerin doğrudan WebSphere MQtarafından

gerçekleştirilememesi gerekir; örneğin, iletilerin işlenmesi için genişletilmiş bir ileti seçimi sağlayıcısı (örneğin, IBM Integration Bus) gerekir.

When an application publishes on a topic string, where one or more subscribers have a selection string selecting on the content of the message, WebSphere MQ will request that the extended message selection provider parse the publication and inform WebSphere MQ whether the publication matches the selection criteria specified by each subscriber with a content filter.

Genişletilmiş ileti seçimi sağlayıcısı, yayının abonenin seçim dizgisiyle eşleştiğini belirtiyorsa, ileti aboneye teslim edilmeye devam eder.

Genişletilmiş ileti seçimi sağlayıcısı, yayının eşleşmediğini belirlerse, ileti aboneye teslim edilmez. Bu, MQPUT ya da MQPUT1 çağrısının, MQRC_PUBLICATION_FAILURE neden kodu ile başarısız olmasına neden olabilir. Genişletilmiş ileti seçimi sağlayıcısı yayını ayrıştıramıyorsa, neden kodu MQRC_CONTENT_ERROR döndürülür ve MQPUT ya da MQPUT1 çağrısı başarısız olur.

Genişletilmiş ileti seçimi sağlayıcısı kullanılamıyorsa ya da abonenin yayını alması gerekip gerekmediğini belirleyemezse, MQRC_SELECTION_NOT_AVAILABLE neden kodu döndürülür; MQPUT ya da MQPUT1 çağrısı başarısız olur.

Bir içerik süzgeci ile bir abonelik yaratılırken ve genişletilmiş ileti seçimi sağlayıcısı yoksa, MQSUB çağrısı MQRC_SELECTION_NOT_AVAM neden kodu ile başarısız olur. İçerik süzgeci olan bir abonelik sürdürülüyorsa ve genişletilmiş ileti seçimi sağlayıcısı yoksa, MQSUB çağrısı MQRC_SELECTION_NOT_AVAM uyarısını döndürür, ancak aboneliğin sürdürülmesine izin verilir.

İlgili kavramlar

Seçim davranışı
IBM WebSphere MQ seçim davranışına genel bakış.

İleti seçici sözdizimi
WebSphere MQ ileti seçici, SQL92 koşullu ifade sözdiziminin bir alt kümesini temel alan sözdizimine sahip bir dizgidir.

IBM WebSphere MQ iletilerinin zamanuyumsuz tüketimi

Zamanuyumsuz tüketim, bir MQI (Message Queue Interface; İleti Kuyruğu Arabirimi) uzantıları kümesini kullanır; MQI, MQCB ve MQCTL ' yi çağırır. Bu, bir MQI uygulamasının iletileri bir kuyruk kümesinden tüketmek üzere yazılmasına olanak sağlar. İletiler, uygulamanın iletiyi ileterek ya da iletiyi gösteren bir belirteç tarafından tanımlanan 'kod birimini' çağırarak uygulamaya teslim edilir.

Uygulama ortamlarında en basit şekilde, 'kod birimi' bir işlev işaretçisi tarafından tanımlanır, ancak diğer ortamlarda 'kod birimi' bir program ya da modül adıyla tanımlanabilir.

İletilerin zamanuyumsuz olarak tüketilmesinde, aşağıdaki terimler kullanılır:

İleti tüketicisi
Uygulamalar gereksinimle eşleşen bir program olduğunda bir ileti ile çağrılacak bir program ya da işlev tanımlamanıza olanak sağlayan bir programlama yapısı.

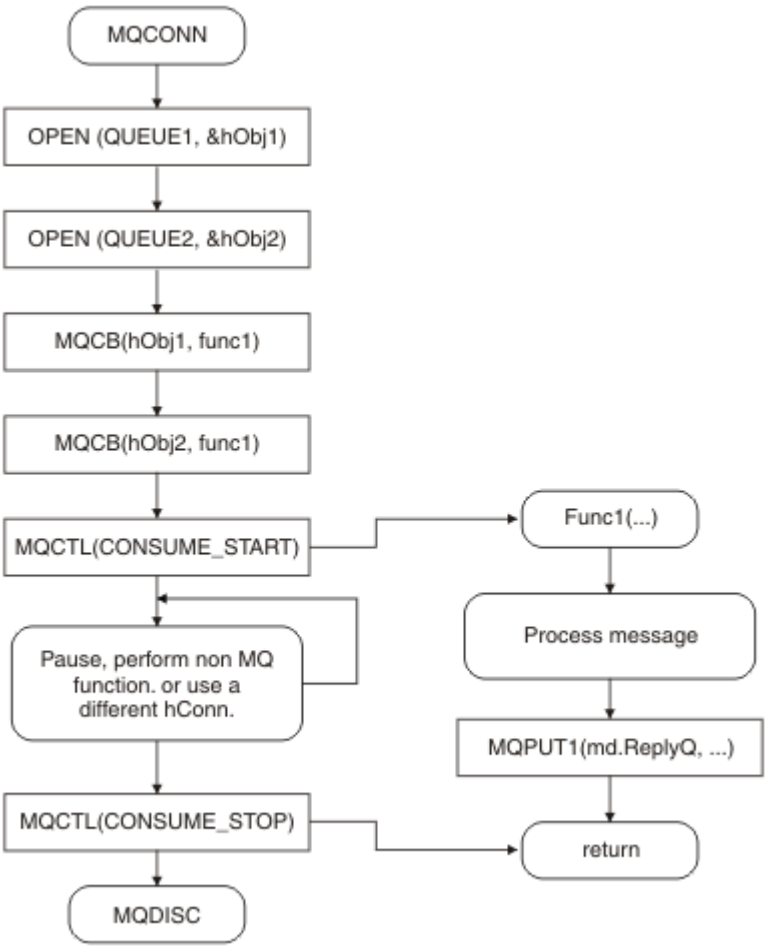
Olay işleyici
Kuyruk yöneticisi susturulması gibi zamanuyumsuz bir olay gerçekleştiğinde çağrılacak bir program ya da işlev tanımlamanıza olanak sağlayan bir programlama yapısı.

Geri çağırma
Bir Message Consumer ya da Event Handler yordamlarına gönderme yapmak için kullanılan soysal terim.

Zamanuyumsuz tüketim, özellikle birden çok giriş kuyruğu ya da aboneliği işleyen yeni uygulamaların tasarlanmasını ve uygulanmasını basitleştirebilir. Ancak, birden fazla giriş kuyruğu kullanıyorsanız ve iletileri öncelik sırasına göre işleiyorsanız, her bir kuyruk içinde öncelik sırası bağımsız olarak görülür: Yüksek öncelikli iletilerin bir kuyruğundan başka bir kuyruktan düşük öncelikli iletiler alabilirsiniz. Birden çok kuyrukta ileti sırası garanti edilmez. Ayrıca, API çıkışlarını kullanırsanız, MQCB ve MQCTL çağrılarını içermek için bunları değiştirmeniz gerekebilir.

Aşağıdaki şekillerde, bu işlevi nasıl kullanabileceğinin bir örneği yer almanıza yardımcı olur.

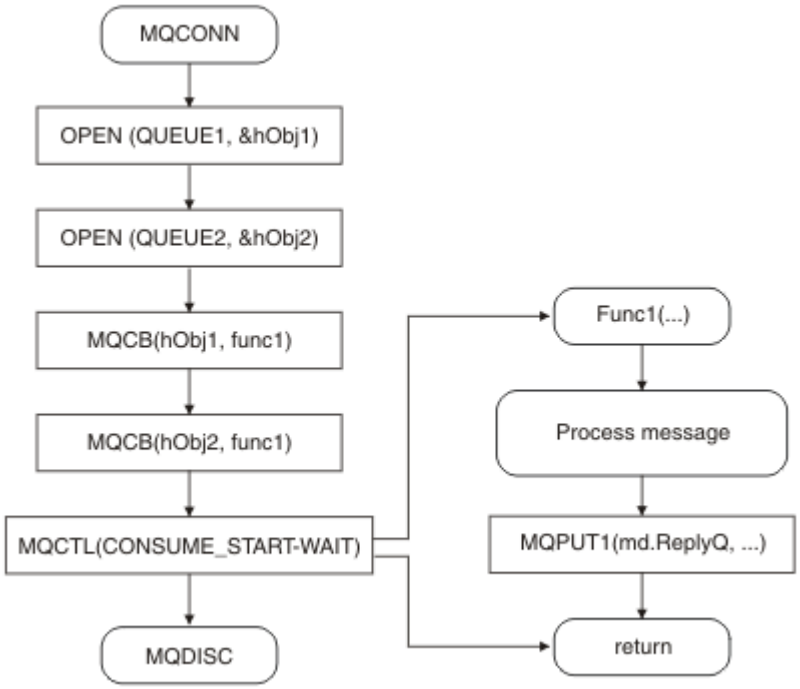
Şekil 5 sayfa 33 , iki kuyruktan gelen iletileri tüketen çok iş parçacıklı bir uygulamayı gösterir. Örnekte, tek bir işleve teslim edilen tüm iletiler gösterilir.



Şekil 5. İki kuyruktan alan standart ileti odaklı uygulama

Şekil 6 sayfa 34 Bu örnek akış, iki kuyruktan gelen iletileri tüketen tek bir iş parçacıklı uygulamayı gösterir. Örnekte, tek bir işleve teslim edilen tüm iletiler gösterilir.

Zamanuyumsuz vakadan farkı, tüm tüketicilerin kendilerini devre dışı bırakılana kadar, denetim MQCTL ' nin yayınına dönmemesi; bu bir tüketici, bir MQCTL STOP isteği ya da kuyruk yöneticisi susturma işlemi yayınlamalıdır.



Şekil 6. İki kuyruktan alan tek iş parçacıklı ileti odaklı uygulama

İleti grupları

İletiler, ileti sıralamasına izin vermek için gruplar içinde oluşabilir.

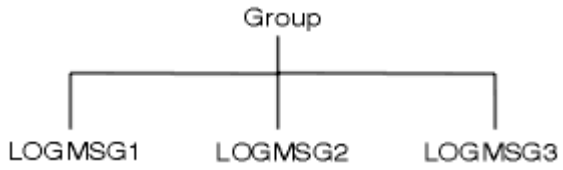
İleti grupları, birden çok iletinin birbiriyle ilişkili olarak imlenmesine ve gruba uygulanmasına ilişkin mantıksal bir sıralamaya izin verir (bkz. "Mantıksal ve fiziksel sıralama" sayfa 235). On platforms other than z/OS, a related concept, "İleti bölümlemesi" sayfa 251 enables large messages to be broken up into smaller segments. Bir konuya yerleştirilirken gruplanmış ya da bölümlenmiş iletileri kullanamazsınız.

Bir grup içindeki sıradüzeni aşağıdaki gibidir:

Grup
Bu, sıradüzendeki en yüksek düzeydir ve bir *GroupId*ile tanımlanır. Aynı *GroupId*' i içeren bir ya da daha fazla iletinden oluşur. Bu iletiler kuyruğun herhangi bir yerinde saklanabilir.

Not: *ileti* terimi, kuyruklardaki bir öğeyi göstermek için burada kullanılır; örneğin, MQGMO_COMPLETE_MSG belirtmeyen tek bir MQGET tarafından döndürülecekti.

Şekil 7 sayfa 34 içinde, bir grup mantıksal ileti gösterilir:



Şekil 7. Mantıksal ileti grubu

Bir kuyruk açıp MQOO_BIND_ON_GROUP belirtilerek, bu kuyruğa gönderilen bir gruptaki tüm iletileri, kuyruğun aynı örneğine gönderilmek üzere zorlayın. BIND_ON_GROUP seçeneği ile ilgili ek bilgi için İleti zenginliklerine işlemebaşlıklı konuya bakın.

Mantıksal ileti
Bir grup içindeki mantıksal iletiler, *GroupId* ve *MsgSeqNumber* alanları tarafından tanımlanır. *MsgSeqNumber* , bir grup içindeki ilk ileti için 1 'den başlar ve bir ileti bir grupta değilse, alanın değeri 1 'dir.

Bir grup içindeki mantıksal iletileri aşağıdaki gibi kullanın:

- Siparişin verildiğinden emin olun (bu, iletinin iletilmekte olduğu koşullar altında garanti verilmezse).
- Uygulamaların benzer iletileri gruplamasına izin ver (örneğin, tümü aynı sunucu eşgörünümü tarafından işlenmek zorunda olanlar).

Bir grup içindeki her ileti, kesimlere bölünmediği sürece, tek bir fiziksel iletten oluşur. Her ileti mantıksal olarak ayrı bir iletidir ve yalnızca MQMD ' deki *GroupId* ve *MsgSeqNumber* alanlarının gruptaki diğer iletilerlerle herhangi bir ilişkiye gereksinimi vardır. MQMD ' deki diğer alanlar bağımsızdır; bazıları, gruptaki tüm iletiler için aynı olabilir, ancak diğerleri farklı olabilir. Örneğin, bir gruptaki iletiler farklı biçim adları, CCSID ' ler ve kodlamalar olabilir.

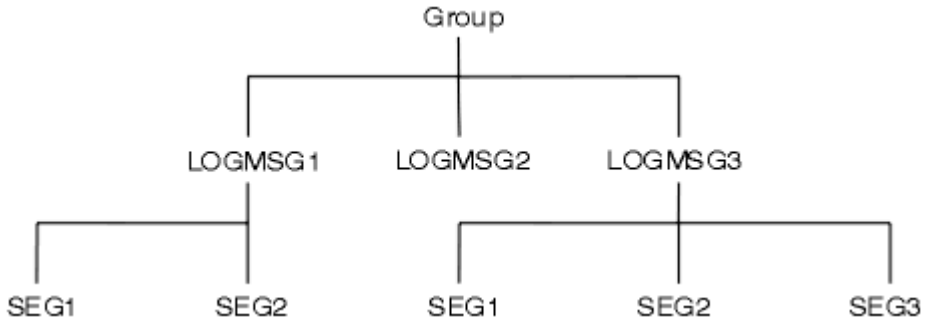
Bölüm

Kesimler, uygulama ya da alma uygulaması ya da kuyruk yöneticisi (iletinin geçtiği, araya giren kuyruk yöneticileri de içinde olmak üzere) için çok büyük olan iletileri işlemek için kullanılır. Daha fazla bilgi için “İleti bölümlemesi” sayfa 251 başlıklı konuya bakın.

Bir ileti, *kesimler*adı verilen daha küçük iletilere bölünmesi halinde ayrılır. A segment of a message is identified by the *GroupId*, *MsgSeqNumber*, and *Offset* fields. *Offset* alanı, bir ileti içindeki ilk bölüm için sıfır düzeyinde başlar.

Her bir kesim, bir gruba ait olabilecek bir fiziksel iletimden oluşur (Şekil 8 sayfa 35 , bir grup içindeki iletilere bir örnek gösterir). Bölüm, mantıksal olarak tek bir iletinin bir parçasıdır; bu nedenle, MQMD ' deki *MsgId*, *Offset*ve *SegmentFlag* alanlarının aynı iletinin ayrı kesimleri arasında farklılık göstermeleri gerekir. Bir kesim gelmezse, neden kodu MQRC_INCOMPLE_GROUP ya da MQRC_INCOMPLE_MSG uygun olarak döndürülür.

Şekil 8 sayfa 35 içinde bir grup mantıksal ileti gösterilir; bunlardan bazıları bölümlenmiş:



Şekil 8. Kesimlere ayrılmış iletiler

Yayınlama/Abone Olma ile bölümlenmiş ya da gruplanmış iletiler kullanamazsınız.

Mantıksal ve fiziksel iletilere ilişkin açıklamalar için bkz. “Mantıksal ve fiziksel sıralama” sayfa 235. İletilere ayrıma hakkında daha fazla bilgi için bkz. “İleti bölümlemesi” sayfa 251.

İleti kalıcılığı

Kalıcı iletiler günlüklere ve kuyruk veri dosyalarına yazılır.

Bir kuyruk yöneticisi bir hatadan sonra yeniden başlatılırsa, günlüğe kaydedilen verilerden gerekli olan bu kalıcı iletileri kurtarır. Kuyruk yöneticisi durursa, kalıcı olmayan iletiler atılır; durdurma sayfası bir işletmen komutunun sonucu olarak mı, yoksa sisteminizin bir kısmının başarısızlığı nedeniyle mi atılır.

Bir ileti yarattığınızda, varsayılan değerleri kullanarak ileti tanımlayıcısını (MQMD) kullanıma hazırlarken, iletiye ilişkin kalıcılık, MQOPED komutunda belirtilen kuyruğun *DefPersistence* özniteliğinden alınır. Diğer bir seçenek olarak, iletiyi kalıcı ya da kalıcı olmayan bir ileti olarak tanımlamak için MQMD yapısının *Persistence* alanını kullanarak iletinin kalıcıtısını ayarlayabilirsiniz.

Kalıcı iletiler kullandığınızda uygulamanızın performansı etkilenir; etkinin kapsamı, makinenin G/Ç altsisteminin başarım özelliklerine ve her altyapıda eşitleme noktası seçeneklerini nasıl kullandığınızı gösterir:

- Geçerli iş biriminin dışında, sürekli bir ileti, her put ve get işleminde diske yazılıdır. Bkz. “İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308.
- In IBM WebSphere MQ on UNIX systems, IBM WebSphere MQ on Linux systems, and IBM WebSphere MQ for Windows, a persistent message within the current unit of work is logged only when the unit of work is committed (and the unit of work could contain many queue operations).

Hızlı ileti sistemi için kalıcı olmayan iletiler kullanılabilir. Hızlı iletiler hakkında daha fazla bilgi için bkz. İletilerin güvenliği .

Not: Bir iş birimi içindeki kalıcı iletilerin yazılması ve kalıcı iletilerin bir birimin ya da işin dışında yazılması, uygulamalarınız için ciddi başarım sorunlarına neden olabilir. Bu durum özellikle, her iki işlem için de aynı hedef kuyruk kullanıldığında geçerlidir.

Teslim edilememiş olan iletiler

Kuyruk yöneticisi bir iletiyi kuyruğa koyamadığında, çeşitli seçenekleriniz vardır.

Yapabilecekleriniz:

- İletiyi yeniden kuyruğa koyma girişiminde bulunuldu.
- İletinin gönderene döndürülemesini isteyin.
- Mesajı ölü mektup kuyruğuna koyun.

Ek bilgi için “Program hatalarının işlenmesi” sayfa 526 başlıklı konuya bakın.

Yedeklenen iletiler

Bir iş biriminin denetimi altındaki bir kuyruktan ileti işlenirken, iş birimi bir ya da daha çok iletiden oluşabilir. Bir geriletme ortaya çıkarsa, kuyruktan alınan iletiler kuyrukta yeniden başlatılır ve başka bir iş biriminde işlenebilir. Belirli bir iletinin işlenmesi soruna neden oluyorsa, iş birimi yeniden yedeklenir. Bu işlem bir işleme döngülerine neden olabilir. Kuyruğa yerleştirilecek iletiler kuyruktan kaldırılır.

Uygulama, MQMD ' nin *BackoutCount* alanını sınayarak böyle bir döngüye yakalanan iletileri saptayabilir. Uygulama durumu düzeltebilir ya da bir operatöre uyarı yayınlayabilir.

On WebSphere MQ for WebSphere MQ for Pencereler, WebSphere MQ on UNIX systems, WebSphere MQ on Linux systems the backout count always survives restarts of the queue manager. *HardenGetBackout* özniteliğe yapılan herhangi bir değişiklik yoksayılır.

İletilerin kesinleştirilmesi ve yedeklenmesi hakkında daha fazla bilgi için bkz. “İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308.

Yanıtın gönderileceği kuyruk ve kuyruk yöneticisi

Gönderdiğiniz bir iletiye yanıt olarak ileti alabileceğiniz durumlar da vardır:

- Bir istek iletisinde yanıt olarak yanıt iletisi
- Beklenmeyen bir olay ya da süre bitimi ile ilgili bir rapor iletisi
- Bir COA (rakip Onaylama Onayı) veya bir COD (Teslim Onayı) olayı hakkında bir rapor iletisi
- Bir PAN (Pozitif İşlem Bildirimi) ya da bir NAN (Negatif İşlem Bildirimi) olayı ile ilgili bir rapor iletisi

Using the MQMD structure, specify the name of the queue to which you want reply and report messages sent, in the *ReplyToQ* field. *ReplyToQMgr* alanında, yanıtın gönderileceği kuyruk yöneticisinin adını belirtin.

ReplyToQMgr alanını boş bırakırsanız, kuyruk yöneticisi, kuyrukta bulunan ileti tanımlayıcısında aşağıdaki alanların içeriğini ayarlar:

ReplyToQ
ReplyToQ uzak bir kuyruğun yerel tanığıysa, *ReplyToQ* alanı uzak kuyruğun adına ayarlanır; tersi durumda bu alan değişmez.

ReplyToQMgr
ReplyToQ uzak bir kuyruğun yerel tanığıysa, *ReplyToQMgr* alanı, uzak kuyruğa sahip olan kuyruk yöneticisinin adına ayarlanır; tersi durumda, *ReplyToQMgr* alanı, uygulamanızın bağlı olduğu kuyruk yöneticisinin adına ayarlanır.

Not: Bir kuyruk yöneticisinin bir iletiyi teslim etmek için birden fazla girişimde bulunmasını isteyebilirsiniz ve hata başarısız olursa iletinin atıldığını isteyebilirsiniz. İleti, teslim edilemedikten sonra atılmayacaksa, uzak kuyruk yöneticisi iletiyi ölüme ilişkin ileti (teslim edilmemiş ileti) kuyruğuna koyar (bkz. "Ölü harf (teslim edilmemiş ileti) kuyruğunun kullanılması" sayfa 529).

İleti bağlamı

İleti bağlamı bilgileri, iletiyi alan uygulamanın, iletiyi oluşturan iletiyi bulmasını sağlayan bir uygulamaya olanak tanır.

Alma uygulaması aşağıdakileri yapmak isteyebilirler:

- Gönderme uygulamasının doğru yetki düzeyine sahip olup olmadığını denetleyin
- Bazı muhasebe işlevini, gerçekleştirmesi gereken işler için gönderme uygulamasını şarj edebilmesini sağlar.
- Üzerinde çalıştığı tüm iletilerin bir denetim izini alıkoy

Bir iletiyi kuyruğa koymak için MQPUT ya da MQPUT1 çağrısını kullandığınızda, kuyruk yöneticisinin ileti tanımlayıcısına varsayılan bağlam bilgileri eklemesini belirleyebilirsiniz. Uygun yetki düzeyine sahip uygulamalar ek bağlam bilgileri ekleyebilir. Bağlam bilgilerinin nasıl belirtileceği hakkında daha fazla bilgi için bkz. "Bağlam bilgilerini denetleme" sayfa 222.

Kullanıcı bağlamı, aşağıdaki rapor iletisi türleri oluşturulurken kuyruk yöneticisi tarafından kullanılır:

- Teslim edilmeyi onayla
- Son kullanma tarihi

Bu rapor iletileri oluşturulduğunda, kullanıcı bağlamı, raporun hedefi üzerinde + put ve + passid yetkisi olup olmadığını denetlenir. Kullanıcı bağlamının yetkisi yetersiz olduğu durumlarda, rapor iletisi, tanımlanmış olan ileti kuyruğuna yerleştirilir. Ölü-mektup kuyruğu olmadığı durumlarda, rapor iletisi atılır.

Tüm bağlam bilgileri, ileti tanımlayıcısının bağlam alanlarında saklanır. Bilgi tipi, kimlik, kaynak ve kullanıcı bağlamı bilgilerine rastlar.

Kimlik bağlamı

Tanıtıcı bağlamı bilgileri, iletiyi ilk olarak kuyruğa koyan uygulamanın kullanıcısını tanımlar. Uygun şekilde yetkili uygulamalar aşağıdaki alanları ayarlayabilir:

- Kuyruk yöneticisi, *UserIdentifier* alanını, kullanıcıyı tanımlayan bir adla doldurur; kuyruk yöneticisinin bunu yapabilme biçimi, uygulamanın çalışmakta olduğu ortama bağlıdır.
- Kuyruk yöneticisi, *AccountingToken* alanını, iletiyi koyan uygulamadan saptadığı bir simgeyle ya da sayıyla doldurur.
- Uygulamalar, kullanıcı hakkında içermek istedikleri ek bilgiler (örneğin, şifrelenmiş bir parola) için *ApplIdentityData* alanını kullanabilir.

A Pencereler systems security identifier (SID) is stored in the *AccountingToken* field when a message is created under WebSphere MQ for Pencereler. SID, *UserIdentifier* alanını tamamlamak ve bir kullanıcının kimlik bilgilerini oluşturmak için kullanılabilir.

Kuyruk yöneticisinin *UserIdentifier* ve *AccountingToken* alanlarını doldurduyla ilgili bilgi için UserIdentifier ve AccountingTokeniçindeki bu alanların tanımlarına bakın.

Bir kuyruk yöneticisinden diğerine ileti geçiren uygulamalar, diğer uygulamaların iletinin kökeninin kimliğini bilmesi için kimlik bağlamı bilgilerini de iletmelidir.

Kaynak bağlamı

Kaynak bağlamı bilgileri, iletiyi *şu anda* ' in saklandığı kuyruğa koyan uygulamayı açıklar. İleti tanımlayıcısı, kaynak bağlamı bilgileri için aşağıdaki alanları içerir:

PutApplType	İletiyi koyan uygulamanın tipi (örneğin, bir CICS işlemi).
PutApplName	İletiyi koyan uygulamanın adı (örneğin, bir işin ya da işlemin adı).
PutDate	İletinin kuyruğa konacağı tarih.
PutTime	İletinin kuyruğa konacağı saat.
ApplOriginData	Bir uygulamanın iletinin kökeniyle ilgili olarak eklemek istediği ek bilgiler. Örneğin, kimlik verilerinin güvenilir olup olmadığını göstermek için uygun yetkili uygulamalar tarafından ayarlanabilirdi.

Kaynak bağlamı bilgileri genellikle kuyruk yöneticisi tarafından sağlanır. Greenwich Ortalama Saati (GMT), *PutDate* ve *PutTime* alanları için kullanılır. PutDate ve PutTimeiçinde bu alanların açıklamalarını görebilirsiniz.

Yeterli yetkiye sahip bir uygulama kendi bağlamını sağlayabilir. Bu, tek bir kullanıcının, kaynaklandığı bir iletiyi işleyen her bir sistemde farklı bir kullanıcı kimliğine sahip olduğunda, muhasebe bilgilerinin korunmasını sağlar.

WebSphere MQ nesneleri

Bu bilgiler, kuyruk yöneticileri, kuyruk paylaşım grupları, kuyruklar, denetim konusu nesneleri, ad listeleri, süreç tanımlamaları, kimlik doğrulama bilgileri nesneleri, kanallar, depolama sınıfları, dinleyiciler ve hizmetler gibi çeşitli WebSphere MQ nesnelerine ilişkin ayrıntıları içerir.

Kuyruk yöneticileri, bu nesnelerin özelliklerini (öznitelikler olarak bilinir) tanımlar. Bu özniteliklerin değerleri, WebSphere MQ ' un bu nesneleri işlemesinin yolunu etkiler. Uygulamalarınızdan bu nesneleri denetlemek için Message Queue Interface (MQI) olanağını kullanıyorsunuz. Nesneler, bir programdan adreslendiklerinde bir *nesne tanımlayıcısı* (MQOD) ile tanımlanır.

Örneğin, nesneleri tanımlamak, değiştirmek ya da silmek için WebSphere MQ komutları kullandığınızda, kuyruk yöneticisi bu işlemleri gerçekleştirmek için gereken yetki düzeyiyle ilgili olup olmadığını denetler. Benzer şekilde, bir uygulama bir nesneyi açmak için MQOPEN çağrısını kullandığında, kuyruk yöneticisi, uygulamanın o nesneye erişime izin verebilmesi için gerekli düzeyde yetki olduğunu denetler. Çekler, açılmakta olan nesnenin adı üzerinde yapılır.

İlgili kavramlar

“Bağlam bilgilerini denetleme” sayfa 222
Bir iletiyi kuyruğa koymak için MQPUT ya da MQPUT1 çağrısını kullandığınızda, kuyruk yöneticisinin ileti tanımlayıcısına varsayılan bağlam bilgileri eklemesini belirleyebilirsiniz. Uygun yetki düzeyine sahip uygulamalar ek bağlam bilgileri ekleyebilir. Bağlam bilgilerini denetlemek için, MQPMO yapısındaki seçenekler alanını kullanabilirsiniz.

İlgili başvurular

“İleti bağlamına ilişkin MQAç seçenekleri” sayfa 213
Bir iletiyi bir kuyruğa koyduğunuzda bağlam bilgilerini bir iletiyle ilişkilendirebilmek istiyorsanız, kuyruğu açtığınızda ileti bağlamı seçeneklerinden birini kullanmanız gerekir.

Microsoft Transaction Server uygulamalarının hazırlanması ve çalıştırılması

Bir MTS uygulamasını WebSphere MQ MQI istemci uygulaması olarak çalıştırmak üzere hazırlamak için, bu yönergeleri ortamınız için uygun olan yönergeleri izleyin.

For general information about how to develop Microsoft Transaction Server (MTS) applications that access WebSphere MQ resources, see the section on MTS in the WebSphere MQ Help Center.

Bir MTS uygulamasını WebSphere MQ MQI istemci uygulaması olarak çalıştırmak üzere hazırlamak için, uygulamanın her bileşeni için aşağıdakilerden birini yapın:

- Bileşen, MQI için C dili bağ tanımlarını kullanıyorsa, “Windows 'ta C programlarının hazırlanması” sayfa 440 içindeki yönergeleri izleyin, ancak bileşeni mqic.libyerine mqicxa.lib kitaplığıyla bağlantılayın.
- Bileşen, WebSphere MQ C++ sınıflarını kullanıyorsa, “C++ programlarını Windowsüzerinde oluşturma” sayfa 627 içindeki yönergeleri izleyin, ancak bileşeni imqc23vn.libyerine imqx23vn.lib kitaplığıyla bağlantılayın.
- Bileşen, MQI için Visual Basic dili bağ tanımlarını kullanıyorsa, “Visual Basic Programlarinin Windows 'ta Hazırlanması” sayfa 443 içindeki yönergeleri izleyin, ancak Visual Basic projesini tanımlarken **Koşullu Derleme Bağımsız Değişkenleri** alanında `MqType=3` yazın.
- If the component uses the WebSphere MQ Automation Classes for ActiveX (MQAX), define an environment variable, GMQ_MQ_LIB, with the value `mqic32xa.dll` .

  Ortam değişkenini uygulamanızın içinden tanımlayabilir ya da kapsamı sistem çapında olacak şekilde tanımlayabilirsiniz. Ancak, sistemi geniş olarak tanımlamak, var olan MQAX uygulamasının, uygulama içinden ortam değişkenini tanımlamamasına, yanlış şekilde davranmasına neden olabilir.

IBM WebSphere MQ ile WebSphere Application Serverkomutunu kullanma

IBM WebSphere MQ ile WebSphere Application Serverkullanımını anlamak için bu konuyu kullanın.

Applications written in Java that are running under WebSphere Application Server can use the Java Messaging Service (JMS) specification to perform messaging. Bu ortamda noktadan noktaya ileti alışverişi bir IBM WebSphere MQ kuyruk yöneticisi tarafından sağlanabilir

Noktadan noktaya ileti sistemini sağlamak için bir IBM WebSphere MQ kuyruk yöneticisi kullanılmasının yararı, JMS uygulamalarının bir IBM WebSphere MQ ağının işlevselliğine tam olarak katılabilmesi ve uygulamaların çok sayıda altyapıda çalışan kuyruk yöneticileriyle ileti alışverişi yapılmasına olanak sağlayan bir avantajdır.

Uygulamalar, kuyruk bağlantısı üreticisi nesnesi için *istemci iletimi* ya da *bağ tanımları aktarımı* ' yı kullanabilir. *Bağ tanımları aktarımı* için, kuyruk yöneticisinin bir bağlantı gerektiren uygulamada yerel olarak varolması gerekir. Kuyruk yöneticisi uygulama için yerel değilse, uygulamanın başka bir makinede ya da görüntüde çalışan bir kuyruk yöneticisine bağlanmasına izin vermek için *İstemci Eki* kurulmalıdır.

Varsayılan olarak, IBM WebSphere MQ kuyruklarında tutulan JMS iletileri, JMS iletisi üstbilgisi bilgilerinin bir kısmını tutmak için bir MQRFH2 üstbilgisini kullanır. Birçok eski IBM WebSphere MQ uygulaması bu üstbilgileri içeren iletileri işleyemez ve kendi karakteristik üstbilgilerini (örneğin, CICS Bridge için MQCIH ya da IBM WebSphere MQ Workflow uygulamaları için MQWIH) gerektiremez. Bu özel konularla ilgili daha fazla ayrıntı için bkz. “JMS iletilerinin WebSphere MQ iletilerine eşlenmesi” sayfa 778.

Hareket desteği senaryoları

İşlem desteğini kullanarak, uygulamalarınızın veritabanlarıyla güvenilir bir şekilde çalışmasını sağlayabilirsiniz.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bu bölümde işlemsel destek tanıtılır. Uygulamalarınızın bir veritabanı ürünüyle IBM WebSphere MQ ' i kullanmasını sağlamak için gereken çalışma, uygulama programlama ve sistem denetimi alanlarına yayılır. Bu bilgileri burada “İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308ile birlikte kullanın.

İşlemleri oluşturan iş birimlerini tanıtarak başlıyoruz, daha sonra IBM WebSphere MQ ' un veritabanlarıyla işlemleri koordine etmek için etkinleştirdiğiniz yolları açıklayarak başlarız.

İlgili kavramlar

"İş birimlerinin tanıtılması" sayfa 40
Bu konu, iş birimi, kesinleştirme, geri alma ve eşitleme noktası genel kavramlarını tanıtır ve tanımlar. Ayrıca, genel çalışma birimlerini gösteren iki senaryo da içerir.

IBM WebSphere MQ ve HP NonStop TMF

İş birimlerinin tanıtılması

Bu konu, iş birimi, kesinleştirme, geri alma ve eşitleme noktası genel kavramlarını tanıtır ve tanımlar. Ayrıca, genel çalışma birimlerini gösteren iki senaryo da içerir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bir program bir iş birimi içindeki kuyruklara ileti yerleştirdiğinde, bu iletiler yalnızca iş birimini *kesinleştirirken* diğer programlar tarafından görülebilir hale gelir. Bir iş birimini kesinleştirmek için, veri bütünlüğünü korumak için tüm güncellemelerin başarılı olması gerekir.

Program bir hata saptarsa ve koyma işlemini kalıcı yapmamaya karar verirse, iş birimi *geri al* olabilir. Bir program bir geriletme gerçekleştirdiğinde, WebSphere MQ , bu iş birimi tarafından kuyruklara konulan iletileri kaldırarak kuyrukları geri yükler.

Benzer bir şekilde, program bir iş birimi içindeki bir ya da daha çok kuyruktan ileti aldığında, program iş birimini kesinleştirinceye kadar bu iletiler kuyruklarda kalır, ancak diğer programlar tarafından alınmak için bu iletiler kullanılamaz. Bu iletiler, program iş birimini kesinleştirirken kuyruklardan kalıcı olarak silinir. Program iş birimini yedeklerse, WebSphere MQ diğer programlar tarafından alınabilebilecek iletileri yaparak kuyrukları geri yükler.

Değişikliklerin kesinleştirme ya da geri alınması, en basit durumda, bir görevin sonunda alınır. Ancak, bir uygulamanın veri değişikliklerini bir görev içindeki diğer mantıksal noktalarda eşitlemesi daha kullanışlı olabilir. Bu mantıksal noktalar, eşzamanlama noktaları (ya da eşitleme noktaları) olarak adlandırılır ve iki eşitleme noktası arasındaki bir güncelleme kümesinin işleme süresi *çalışma birimi*olarak adlandırılır. Birden çok MQGET çağrısı ve MQPUT çağrıları tek bir iş biriminin bir parçası olabilir.

WebSphere MQile, iş birimlerinin *yerel* ile *genel* birimlerini birbirinden ayırt etmek gerekir:

Yerel iş birimleri
Are those in which the only actions are puts to, and gets from, WebSphere MQ queues, and the coordination of each unit of work is provided within the queue manager using a *tek aşamalı kesinleştirme* process.

Güncellenecek kaynaklar tek bir WebSphere MQ kuyruk yöneticisi tarafından yönetilen kuyruklar olduğunda, yerel iş birimlerini kullanın. Güncellemeler, MQCMIT komutu kullanılarak ya da MQBACK kullanılarak yedeklenilerek kesinleştirilir.

Yerel iş birimlerinin kullanılmasına dahil olan günlük yönetimi dışında, sistem denetimi görevi yoktur. MQPUT ve MQGET çağrılarını MQCMIT ve MQBACK ile birlikte kullandığınız uygulamalarınızda, MQPMO_SYNCPOINT ve MQGMO_SYNCPOINT seçeneklerini kullanmayı deneyin. (Günlük yönetimiyle ilgili bilgi için Günlük dosyalarının yönetilmesi başlıklı konuya bakın.)

Genel iş birimleri
İlişkisel veri tabanındaki çizelgeler gibi diğer kaynakların da güncellendiği kişiler olabilir. Birden çok *kaynak yöneticisi* yer aldığında, genel iş birimini koordine etmek için *iki aşamalı kesinleştirme* işlemini kullanan *transaction manager* yazılıma gereksinim vardır.

Use global units of work when you also need to include updates to relational database manager software, such as Db2, Oracle, Sybase, and Informix.

Genel iş birimlerinin kullanılması için bazı olası senaryolar vardır. Burada belgelenmiş iki senaryo vardır:

1. İlk olarak, kuyruk yöneticisi hareket yöneticisi olarak işlev görür. Bu senaryoda, MQI fiilleri iş genel birimlerini denetler; MQBEGIN komutunu kullanan uygulamalarda başlatılır ve MQCMIT kullanılarak kesinleştirilir ya da MQBACK kullanılarak yedeklenirler.
2. İkinci olarak, hareket yöneticisi rolü, TXSeries, Encinaya da Tuxedo gibi diğer yazılımlar tarafından gerçekleştirilir. Bu senaryoda, hareket yöneticisi yazılımı tarafından sağlanan bir API, iş birimini denetlemek için kullanılır (örneğin, EXEC CICS SYNCPOINT FOR TXSeries).

Aşağıdaki bölümlerde, iki senaryoya göre düzenlenmiş, genel iş birimlerinin kullanılması için gerekli olan tüm adımlar açıklanmıştır:

- "Senaryo 1: Kuyruk yöneticisi koordinasyonu gerçekleştirir" sayfa 41
- "2. senaryo: Diğer yazılımlar koordinasyonu sağlar" sayfa 66

Senaryo 1: Kuyruk yöneticisi koordinasyonu gerçekleştirir

Senaryo 1 'de, kuyruk yöneticisi hareket yöneticisi olarak işlev görür. Bu senaryoda, MQI fiilleri iş genel birimlerini denetler; MQBEGIN komutunu kullanan uygulamalarda başlatılır ve MQCMIT kullanılarak kesinleştirilir ya da MQBACK kullanılarak yedeklenirler.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Yalıtma düzeyi

IBM WebSphere MQürününde, veritabanı içinde uygulanan işlem yalıtım tasarımına bağlı olarak, veritabanı güncellemesinden önce bir kuyrukta ileti görünür.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bir IBM WebSphere MQ kuyruk yöneticisi XA hareket yöneticisi olarak çalışırken, güncellemeleri XA kaynak yöneticilerine eşleştirmek için aşağıdaki kesinleştirme protokolünün izlenmesini sağlayın:

1. Tüm XA kaynak yöneticilerini hazırlayın.
2. IBM WebSphere MQ kuyruk yöneticisi kaynak yöneticisini kesinleştirin.
3. Diğer kaynak yöneticilerini kesinleştirin.

2. ve 3. adım arasında, bir uygulama, kuyruğa bağlı bir iletiyi görebilir, ancak veritabanındaki karşılık gelen satır bu iletiyi yansıtmaz.

Veritabanının API çağrıları, bekleyen güncellemelerin tamamlanmasını beklemesi gibi bir veritabanı yapılandırıldıysa, bu bir sorun değildir.

Veritabanını farklı bir şekilde yapılandırarak bu sorunu çözebilirsiniz. Gereken yapılandırma tipi, "yalıtma düzeyi" olarak anılır. Yalıtım düzeyleriyle ilgili daha fazla bilgi için veritabanı belgelerine bakın. Diğer bir seçenek olarak, kaynak yöneticilerini aşağıdaki ters sırada kesinleştirmek için kuyruk yöneticisini yapılandırabilirsiniz:

1. Tüm XA kaynak yöneticilerini hazırlayın.
2. Diğer kaynak yöneticilerini kesinleştirin.
3. IBM WebSphere MQ kuyruk yöneticisi kaynak yöneticisini kesinleştirin.

Protokolü değiştirdiğinizde, IBM WebSphere MQ kuyruk yöneticisi son olarak kesinleştirilir; bu nedenle, kuyruklardan ileti okuyan uygulamalar, ilgili veritabanı güncellemesinin tamamlanmasından sonra yalnızca bir ileti görür.

Kuyruk yöneticisini bu değiştirilmiş protokolü kullanacak şekilde yapılandırmak için, **AMQ_REVERSE_COMMIT_ORDER** ortam değişkenini ayarlayın.

Set this environment variable in the environment from which the **strmqm** is run to start the queue manager. Örneğin, kuyruk yöneticisini başlatmadan önce, kabukta aşağıdaki bilgileri çalıştırın:

```
export AMQ_REVERSE_COMMIT_ORDER=1
```

Not: Bu ortam değişkeninin ayarlanması, hareket başına fazladan bir günlük girişine neden olabilir; bu nedenle, her hareketin başarımı küçük bir etkiye neden olur.

Veritabanı koordinasyonu

Kuyruk yöneticisi, iş birimlerinin genel birimlerini koordine ettiğinde, veritabanı güncellemelerini iş birimleri içinde tümleştirmek mümkün olur. Bu, karma bir MQI ve SQL uygulaması yazılabilir ve kuyruklarda ve veritabanlarında birlikte değişiklikleri kesinleştirmek ya da geri almak için MQCMIT ve MQBACK fiilleri kullanılabilir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Kuyruk yöneticisi, *X/Open Distributed Transaction Processing: XA Belirtimi*içinde açıklanan iki aşamalı kesinleştirme protokolünü kullanarak bunu başarır. Bir iş birimi kesinleştirildiğinde, kuyruk yöneticisi ilk olarak her bir katılımcı veritabanı yöneticisiyle ilgili güncellemeleri kesinleştirmek için hazırlanıp hazırlanmadığını sorar. Yalnızca, kuyruk yöneticisinin kendisi de içinde olmak üzere tüm katılımcılar, kesinleştirme işlemi için hazırlandıysa, tüm kuyruk ve veritabanı güncellemeleri kesinleştirilir. Herhangi bir katılımcı güncelliğini hazırlayamazsa, iş birimi geri çekilmez.

Genel olarak, aşağıdaki yöntem (sözde kodda) tarafından bir uygulamada genel bir çalışma birimi uygulanır:

MQBEGIN
MQGET (ileti seçeneklerinde MQGMO_SYNCPOINT işareti ekleyin)
MQPUT (ileti seçeneklerinde MQPMO_SYNCPOINT işaretini içer)
SQL EKLE
MQCMIT

MQBEGIN amacı, genel bir iş biriminin başlangıcını belirtmek için. MQCMIT ' in amacı, genel iş biriminin sonunu göstermek ve iki aşamalı kesinleştirme protokolünü kullanarak tüm katılımcı kaynak yöneticileriyle birlikte tamamlamayı amaçlatır.

İş birimi (*transaction*olarak da bilinir), MQCMIT kullanılarak başarıyla tamamlandığında, o iş birimi içinde alınan tüm işlemler kalıcı ya da geri çevrilemez hale getirilmektedir. Herhangi bir nedenle, iş birimi başarısız olursa, tüm işlemler geri çekilmek yerine desteklenir. Bir iş birimindeki tek bir işlemin kalıcı olarak yapılması mümkün değildir; bu işlem, başka bir işlem tarafından yedeklenmektedir. Bu, bir iş biriminin ilkesidir: İş birimi içindeki tüm eylemler kalıcı yapılır ya da bunların hiçbiri değildir.

Not:

1. Uygulama programcısı, MQBACK ' ı çağırarak bir iş birimini yedeklemek için zorlayabilir. MQCMIT çağrılmadan önce uygulama ya da veritabanı *başarısız olursa* , iş birimi kuyruk yöneticisi tarafından da desteklenmektedir.
2. Bir uygulama MQCMIT çağrılmadan MQDISC ' yi çağrıyorsa, kuyruk yöneticisi, MQCMIT çağrılmış gibi davranır ve iş birimini kesinleştirir.

MQBEGIN ile MQCMIT arasında, kuyruk yöneticisi, veritabanını kaynaklarını güncellemek için herhangi bir çağrıda bulunmaz. Yani, bir veritabanının çizelgelerinin değiştirmenin tek yolu kodun kodlarıdır (örneğin, sözde koddaki SQL INSERT).

Kuyruk yöneticisi, kesinleştirme protokolü sırasında veritabanı yöneticilerinden herhangi biriyle iletişim kurduysa, tam kurtarma desteği sağlanır. Bir veritabanı yöneticisi, kesinleştirilirken kullanılamaz duruma gelirse, kesinleştirmek için başarıyla hazırlanmış, ancak henüz bir kesinleştirme ya da geri alma kararı almadıysa, kuyruk yöneticisi iş biriminin sonucunu, sonuca başarıyla teslim edilinceye kadar anımsatır. Benzer şekilde, kuyruk yöneticisi tamamlanmamış kesinleştirme işlemleriyle sona erdirilirse, bunlar kuyruk yöneticisi yeniden başlatıldığında hatırlanır. Bir uygulama beklenmedik bir şekilde sona ererse,

iş biriminin bütünlüğü tehlikeye atılmaz, ancak sonuç, Çizelge 5 sayfa 43içinde açıklandığı gibi, uygulamanın işleyişi nerede sona erdirildiyse bağlıdır.

Veritabanı ya da uygulama programı başarısız olduğunda, aşağıdaki tablolarda özetlenirse ne olur:

Çizelge 4. Veritabanı sunucusu başarısız olduğunda ne olur	
Hata ortaya çıktı	**sonuç**
Uygulama MQCMIT ' e çağrılmadan önce.	İş birimi geriletilir.
MQCMIT ' e yapılan uygulama çağrısı sırasında, **önce** tüm veritabanları başarıyla hazırlamış olduklarını belirttiler.	İş birimi, MQRC_BACKED_OUT neden koduyla yedeklenir.
MQCMIT uygulama çağrısı sırasında, **bundan sonra** tüm veritabanları başarıyla hazırlamış olduklarını belirtmiş, ancak tüm bunlar başarıyla kesinleştirdiklerini belirtmiş olabilir.	İş birimi, kuyruk yöneticisi tarafından, MQRC_OUTCOME_PENDING neden kodu ile kurtarılabilir durumda tutulur.
MQCMIT ' e yapılan uygulama çağrısı sırasında, **bundan sonra** tüm veritabanları başarıyla kesinleştirdiklerini belirttiler.	İş birimi, MQRC_NONE bir neden koduyla kesinleştirilir.
Uygulama MQCMIT ' e çağrıldıktan sonra.	İş birimi, MQRC_NONE bir neden koduyla kesinleştirilir.

Çizelge 5. Uygulama programı başarısız olduğunda ne olur	
Hata ortaya çıktı	**sonuç**
Uygulama MQCMIT ' e çağrılmadan önce.	İş birimi geriletilir.
During the application call to MQCMIT, **önce** the queue manager has received the application's MQCMIT request.	İş birimi geriletilir.
MQCMIT uygulama çağrısı sırasında, kuyruk yöneticisi uygulamanın MQCMIT isteğini aldı **sonra** .	Kuyruk yöneticisi, iki aşamalı kesinleştirmeyi (veritabanı ürünlerine bağlı olarak, iş biriminin bölümlerini başarıyla yürütmeyi ve kesinleştirmeyi) kullanarak kesinleştirmeyi dener.

MQCMIT ' den dönüş sırasında neden kodunun MQRC_OUTCOME_PENDING olduğu durumlarda, iş birimi kuyruk yöneticisi tarafından, veritabanı sunucusuyla yeniden iletişim kurabilinceye kadar ve iş biriminin bir parçası olarak kesinleştirmesini gerçekleştirinceye kadar hatırlanır. Kurtarma işlemi nasıl ve ne zaman gerçekleştirilmeye ilişkin bilgi için “Kişi XA kaynak yöneticisiyle kaybolduğunda dikkat edilecek noktalar” sayfa 59 başlıklı konuya bakın.

Kuyruk yöneticisi, *X/Open Distributed Transaction Processing: XA Specification*içinde açıklandığı gibi XA arabirimini kullanarak veritabanı yöneticileriyle iletişim kurar. Bu işlev çağrılarına örnekler şunlardır: xa_open, xa_start, xa_end, xa_prepare ve xa_commit. XA belirtiminde kullanılanlarla aynı anlamda *hareket yöneticisi* ve *kaynak yöneticisi* terimlerini kullanırız.

Kısıtlamalar
Veritabanı koordinasyonu desteğine ilişkin kısıtlamalar vardır.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki kısıtlamalar söz konusudur:

- The ability to coordinate database updates within WebSphere MQ units of work is **değil** supported in an MQI client application. Bir istemci uygulamasında MQBEGIN kullanımı başarısız olur. MQBEGIN 'i çağıran bir program, kuyruk yöneticiyle aynı makinede bir *sunucu* uygulaması olarak çalıştırılmalıdır.

  Not: *Sunucu* uygulaması, gerekli WebSphere MQ sunucusu kitaplıklarıyla bağlantılı bir programdır; bir *istemci* uygulaması, gerekli WebSphere MQ istemcisi kitaplıklarıyla bağlantılı bir programdır. Programlarınızın derlenmesine ve bağlanmaya ilişkin ayrıntılar için “WebSphere MQ MQI istemcilerine ilişkin uygulamalar oluşturuluyor” sayfa 340 ve “IBM WebSphere MQ uygulaması oluşturulması” sayfa 409 başlıklı konuya bakın.
- Veritabanı istemcisi kuyruk yöneticisi ile aynı makinede kurulu olduğu sürece, veritabanı sunucusu kuyruk yöneticisi sunucusundan farklı bir makinede bulunabilir ve bu işlevi destekleyebilir. İstemci yazılımlarının iki aşamalı kesinleştirme sistemleri için kullanılıp kullanılamayacağını belirlemek için veritabanı ürününün belgelerine bakın.
- Kuyruk yöneticisi bir kaynak yöneticisi gibi davransa da (senaryo 2 genel iş birimlerinde yer alma amacıyla), bir kuyruk yöneticisinin senaryosu 1 genel iş birimleri içinde başka bir kuyruk yöneticisini koordine etmesi olanaklı değildir.

Yükleme dosyalarını değiştir

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

The switch load file is a shared library (a DLL on Windows systems) that is loaded by the code in your IBM WebSphere MQ application and the queue manager. Bunun amacı, veritabanının istemci paylaşılan kitaplığının yüklenmesini basitleştirmek ve işaretçileri XA işlevlerine geri döndürmek.

Kuyruk yöneticisi başlatılmadan önce anahtar yükleme dosyasının ayrıntıları belirtilmelidir. Ayrıntılar, Windows, UNIX and Linux sistemlerindeki qm.ini dosyasına yerleştirilir.

- Windows ve Linux (x86 ve x86-64 platformları) sistemlerinde, qm.ini dosyasını güncellemek için IBM WebSphere MQ Explorer dosyasını kullanın.
- Diğer tüm sistemlerde, dosya qm.inidoğrudan düzenlenmektedir.

Anahtar yükleme dosyasına ilişkin C kaynağı, Senaryo 1 genel iş birimlerini destekliyorsa, IBM WebSphere MQ kuruluşuyla birlikte sağlanır. Kaynak, MQStart olarak adlandırılan bir işlev içeriyor. Anahtar yükleme dosyası yüklendiğinde, kuyruk yöneticisi bu işlevi çağırır; bu işlev, *XA anahtarı*adı verilen bir yapının adresini döndürür.

XA anahtar yapısı, veritabanı istemcisi paylaşılan kitaplığında bulunur ve Çizelge 6 sayfa 44içinde açıklandığı gibi bir dizi işlev göstericiyi içerir:

Çizelge 6. XA anahtarı işlev işaretçileri

İşlev göstergesi adı	**XA işlevi**	**Amaç**
xa_open_entry	xa_open	Veritabanına bağlan
xa_close_entry	xa_kapat	Veritabanıyla bağlantıyı kes
xa_start_entry	xa_start	Genel iş biriminin bir dalını başlatır
xa_end_entry	xa_end	Genel iş birimi dalını askıya al
xa_rollback_entry	xa_geriye	Genel bir iş biriminin dalını geri alma
xa_prepare_entry	xa_hazırlama	Küresel bir iş biriminin dalını işlemeye hazır olun
xa_commit_girişi	xa_kesinleştirme	Genel iş biriminin bir dalını kesinleştirin

Çizelge 6. XA anahtarı işlev işaretçileri (devamı var)		
İşlev göstergesi adı	**XA işlevi**	**Amaç**
xa_recover_entry	xa_kurtarma	Veritabanında kuşku içinde bir çalışma birimi olup olmadığını veritabanından öğrenin
xa_unutkan_girişi	xa_unut	Bir veritabanının genel iş birimi dalını unutmasına izin ver
xa_complete_entry	xa_tamamlandı	Genel iş biriminin bir dalını tamamla

Uygulamanınızdaki ilk MQBEGIN çağrısı sırasında, MQBEGIN bir parçası olarak yürütülen IBM WebSphere MQ kodu, anahtar yükleme dosyasını yükler ve veritabanı paylaşılan kitaplığında xa_open işlevini çağırır. Benzer şekilde, kuyruk yöneticisi başlatma sırasında ve sonraki diğer durumlarda, bazı kuyruk yöneticisi işlemleri anahtar yükleme dosyasını yükler ve xa_open çağrısını açar.

*dinamik kayıt*komutunu kullanarak xa_ * çağrılarının sayısını azaltabilirsiniz. Bu eniyileme tekniğiyle ilgili eksiksiz açıklamalar için bkz. “XA dinamik kaydı” sayfa 64.

Sisteminizi veritabanı koordinasyonu için yapılandırma

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Veritabanı yöneticisi, kuyruk yöneticisi tarafından eşgüdümlü olarak genel işlev birimlerine katılmadan önce gerçekleştirmeniz gereken birkaç görev vardır. Bunlar aşağıda anlatıldığı gibi açıklanmaktadır:

- “Veritabanı ürününün kurulması ve yapılandırılması” sayfa 45
- “Anahtar yükleme dosyaları oluşturma” sayfa 46
- “Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi” sayfa 46
- “Uygulamalarınızı yazma ve değiştirme” sayfa 48
- “Sistemin sınanması” sayfa 49

Veritabanı ürününün kurulması ve yapılandırılması
Veritabanı ürününüzü kurmak ve yapılandırmak için ürünün kendi belgelerine bakın. Bu bölümdeki bu konular, genel yapılandırma sorunlarını ve bunların WebSphere MQ ile veritabanı arasındaki etkileşimlerle nasıl ilişkilendirildiklerini açıklar.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Veritabanı bağlantıları

Kuyruk yöneticisine standart bir bağlantı kuran bir uygulama, ayrı bir yerel kuyruk yöneticisi aracısı sürecindeki bir iş parçacıkla ilişkilendirildi. (*fastpath* bağlantısı olmayan bir bağlantı, bu bağlamdaki bir *standart* bağlantıdır. Daha fazla bilgi için bkz. “MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 200.)

When the application issues MQBEGIN, both it and the agent process call the xa_open function in the database client library. In response to this, the database client library code *bağlantı* to the database that is to be involved in the unit of work *hem uygulama, hem de kuyruk yöneticisi süreçlerinden*. Uygulama kuyruk yöneticisine bağlı kaldığı sürece bu veritabanı bağlantıları korunur.

Veritabanı, tek bir uygulama programını desteklemek üzere veritabanında iki bağlantı yapıldığından, yalnızca sınırlı sayıda kullanıcı ya da bağlantı destekliyorsa, bu önemli bir değerlendirmeye sahip olur.

İstemci/sunucu yapılandırması

WebSphere MQ kuyruk yöneticisi ve uygulama işlemlerine yüklenen veritabanı istemcisi kitaplığı, sunucudan gönderme ve alma işlemlerini **yapabilmelidir** . Aşağıdakilere dikkat edin:

- Veritabanının istemci/sunucu yapılandırma dosyaları doğru ayrıntılara sahip
- The relevant environment variables are set in the environment of the queue manager **ve** the application processes

Anahtar yükleme dosyaları oluşturma
WebSphere MQ , desteklenen veritabanı yöneticileri için anahtar yükleme dosyaları oluşturmak için kullanılan bir örnek makefile ile birlikte gönderilir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Anahtar yükleme dosyalarını oluşturmak için gerekli tüm ilişkili C kaynak dosyalarıyla birlikte örnek makefile aşağıdaki dizinlere kurulur:

- WebSphere MQ for Windowsiçin, *MQ_INSTALLATION_PATH*`\tools\c\samples\xatm\` dizininde
- For WebSphere MQ for UNIX and Linux systems, in the *MQ_INSTALLATION_PATH*`/samp/xatm/` directory

Anahtar yükleme dosyalarını oluşturmak için kullanılan örnek kaynak modüller şunlardır:

- DB2için, db2swit.c
- Oracleiçin, oraswit.c
- Informix, infswit.ciçin
- Sybaseiçin, sybswit.c

When you generate switch load files, install 32-bit switch load files in `/var/mqm/exits` and install 64-bit switch load files in `/var/mqm/exits64`.

32 bit kuyruk yöneticileriniz varsa, örnek make file (xaswit.mak), `/var/mqm/exits`içinde 32 bit anahtar yükleme dosyası kurar.

64 bit kuyruk yöneticileriniz varsa, örnek make dosyası (xaswit.mak), `/var/mqm/exits`'ta 32 bit anahtar yükleme dosyası ve `/var/mqm/exits64`' ta 64 bit anahtar yükleme dosyası kurar.

Dosya güvenliği

It is possible that your operating system might fail the loading of the switch load file by WebSphere MQ, for reasons outside the control of WebSphere MQ. Böyle bir durumda, hata iletileri WebSphere MQ hata günlüklerine yazılır ve MQBEGIN çağrısının başarısız olması olabilir. İşletim sisteminizin anahtar yükleme dosyasının yüklenmesini başarısız kılmadığından emin olmak için aşağıdaki gereksinimleri yerine getirmeniz gerekir:

1. Anahtar yükleme dosyası, qm.ini dosyasında verilen yerde kullanılabilir olmalıdır.
2. Anahtar yükleme dosyasının, kuyruk yöneticisi işlemleri ve uygulama işlemleri de içinde olmak üzere, yüklenmesi gereken tüm süreçlere erişebilmeleri gerekir.
3. Anahtar yükleme dosyasının bağlı olduğu tüm kitaplıklarda, veritabanı ürünü tarafından sağlanan kitaplıklar da içinde olmak üzere, bu kitaplıkların bulunması ve erişilebilir olması gerekir.

Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi
Veritabanı yöneticiniz için bir anahtar yükleme dosyası yarattığınız ve bunu güvenli bir yere yerleştirdiğinizde, bu yeri kuyruk yöneticinizin yerine belirtmeniz gerekir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Konumu belirtmek için aşağıdaki adımları gerçekleştirin:

- On Pencereler and Linux (x86 and x86-64 platforms) systems use the WebSphere MQ Explorer. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin.
- Diğer tüm sistemlerde, kuyruk yöneticisinin qm.ini dosyasında yer alan XAResourceManager stanza içindeki anahtar yükleme dosyasının ayrıntılarını belirtin.

Kuyruk yöneticinizin koordine edecek olduğu veritabanı için bir XAResourceManager stanza ekleyin. En yaygın durum, yalnızca tek bir veritabanı olması ve bu nedenle yalnızca bir XAResourceManager stanza olması içindir. Birden çok veritabanı içeren daha karmaşık yapılandırmalara ilişkin ayrıntılar için bkz. “Birden çok veritabanı yapılandırması” sayfa 58. XAResourceManager stanza 'nın öznitelikleri aşağıdaki gibidir:

Ad=ad
Kaynak yöneticisini tanımlayan kullanıcı tarafından seçilen dizgi. Sonuç olarak, XAResourceManager stanza adını verir. Ad zorunludur ve en çok 31 karakter uzunluğunda olabilir.

Seçtiğiniz ad benzersiz olmalıdır; bu qm.ini dosyasında bu adı taşıyan tek bir XAResourceManager kısmı olmalıdır. Kuyruk yöneticisi bu kaynak yöneticisine, hem kuyruk yöneticisi hata günlüğü iletilerinde hem de `dspmqtrn` komutu kullanıldığında çıkışta bu kaynak yöneticisine gönderme yapmak için bu adı kullandığından da anlamlı olmalıdır. (Ek bilgi için “Bekleyen iş birimlerinin dspmqtrn komutu ile görüntülenmesi” sayfa 60 konusuna bakın.)

Bir ad seçtikten sonra kuyruk yöneticisini başlattıktan sonra, Name özniteliğini değiştirmeyin. Yapılandırma bilgilerinin değiştirilmesiyle ilgili daha fazla ayrıntı için bkz. “Yapılandırma bilgilerinin değiştirilmesi” sayfa 62.

SwitchFile= ad
Bu ad, daha önce yaptırdığınız XA anahtar yükleme dosyasının adıdır. Bu zorunlu bir öznitedir. Kuyruk yöneticisinde ve WebSphere MQ uygulama süreçlerindeki kod, anahtar yükleme dosyasını şu iki durumda yüklemeyi dener:

1. Kuyruk yöneticisi başlatma sırasında
2. WebSphere MQ uygulama sürecinizde MQBEGIN ilk çağrısını gerçekleştirdiğinizde

Anahtar yükleme dosyanızın güvenlik ve izin öznitelikleri, bu işlemlerin bu işlemi gerçekleştirmesine izin vermelidir.

XAOpenString= dizgi
Bu, veritabanı yöneticisinin xa_open işlevine yönelik çağrılarında WebSphere MQ kodunun geçtiği bir veri dizgisidir. Bu isteğe bağlı bir öznitedir; atlanırsa sıfır uzunluklu bir dizgi varsayılır.

Kuyruk yöneticisinde ve WebSphere MQ uygulama süreçlerindeki kod, xa_open işlevini iki kez çağır:

1. Kuyruk yöneticisi başlatma sırasında
2. WebSphere MQ uygulama sürecinizde MQBEGIN ilk çağrısını gerçekleştirdiğinizde

Bu dizginin biçimi, her veritabanı ürünü için özeldir ve bu ürüne ilişkin belgelerde anlatılacaktır. Genel olarak, xa_open dizgisi, hem kuyruk yöneticisinde hem de uygulama süreçlerinde veritabanına bağlantı sağlamak için kimlik doğrulama bilgilerini (kullanıcı adı ve parola) içerir.

XACloseString= dizgi
Bu, veritabanı yöneticisinin xa_close işlevine yönelik çağrılarında WebSphere MQ kodunun geçtiği bir veri dizgisidir. Bu isteğe bağlı bir öznitedir; atlanırsa sıfır uzunluklu bir dizgi varsayılır.

Kuyruk yöneticisinde ve WebSphere MQ uygulama süreçlerindeki kod, xa_close işlevini iki kez çağır:

1. Kuyruk yöneticisi başlatma sırasında
2. WebSphere MQ uygulama sürecinizde MQDISC çağrısını gerçekleştirdiğinizde, daha önce MQBEGIN çağrısı yapıldı.

Bu dizginin biçimi, her veritabanı ürünü için özeldir ve bu ürüne ilişkin belgelerde anlatılacaktır. Genel olarak, dizgi boştur ve XAResourceManager stanza içindeki XACloseString özniteliğini atlamak yaygındır.

ThreadOfControl=THREAD |PROCESS

ThreadOfDenetim değeri THREAD ya da PROCESS olabilir. Kuyruk yöneticisi bunu diziselleştirme amacıyla kullanır. Bu isteğe bağlı bir öznitedir; atlanırsa, PROCESS değeri kabul edilir.

Veritabanı istemci kodu, iş parçacıklarının serileştirmeden XA işlevlerini çağırmasına izin veriyorsa, ThreadOfControl (Denetim) için değer THREAD (THREAD) olabilir. Kuyruk yöneticisi, veritabanı istemcisi paylaşılan kitaplığında bulunan XA işlevlerini birden çok iş parçacığının aynı anda çağırabileceğini varsayar; gerekiyorsa.

Veritabanı istemci kodu, iş parçacıklarının XA işlevlerini bu şekilde çağırmasına izin vermiyorsa, ThreadOfControl (Denetim) için değer PROCESS (süreç) olmalıdır. Bu durumda, kuyruk yöneticisi tüm çağrıları veritabanı istemcisi paylaşılan kitaplığına diziselleştirir, böylece belirli bir süreçten yalnızca tek bir çağrıda bulunmanız gerekir. Büyük olasılıkla, birden çok iş parçacığıyla çalışırsa, uygulamanızın benzer serileştirme gerçekleştirdiğinden emin olmanız da gerekir.

Bu sorunun, veritabanı ürününün çok iş parçacıklı süreçlerle başa çıkabilme yetenünün bu şekilde, ürünün satıcısına ilişkin bir sorun olduğuna dikkat edin. ThreadOfControl (Denetim) özniteliğini THREAD ya da PROCESS değerine ayarlayıp ayarlayamayacağınızı ilişkin ayrıntılar için veritabanı ürün belgelerine bakın. ThreadOfControl to THREAD ' i (Control to THREAD) ayarladığınızı, bunu öneriyoruz. If in doubt, the *daha güvenli* option is to set it to PROCESS, although you will lose the potential performance benefits of using THREAD.

Uygulamalarınızı yazma ve değiştirme
Küresel bir iş birimi nasıl uygulanacak?

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bir WebSphere MQ kurulumu ile birlikte sağlanan Senaryo 1 genel çalışma birimlerine ilişkin örnek uygulama programları “İş birimlerinin tanıtılması” sayfa 40içinde açıklanmıştır.

Genel olarak, aşağıdaki yöntem (sözde kodda) tarafından bir uygulamada genel bir çalışma birimi uygulanır:

```
MQBEGIN
MQGet
MQPUT
SQL EKLE
MQCMIT
```

MQBEGIN amacı, genel bir iş biriminin başlangıcını belirtmek için. MQCMIT ' in amacı, genel iş biriminin sonunu göstermek ve iki aşamalı kesinleştirme protokolünü kullanarak tüm katılımcı kaynak yöneticileriyle birlikte tamamlamayı amaçlatır.

MQBEGIN ile MQCMIT arasında, kuyruk yöneticisi, veritabanını kaynaklarını güncellemek için herhangi bir çağrıda bulunmaz. Yani, bir veritabanının çizelgelerinin değiştirmenin tek yolu kodun kodlarıdır (örneğin, sözde koddaki SQL INSERT).

Kuyruk yöneticisinin rolü, veritabanı endişeli olduğu sürece, genel iş birimi sona erdiğinde, genel iş biriminin kesinleştirilip işlenmeyeceğini ya da geriye işlenip işlenmeyeceğini ya da geri döndürülediğinde bunu anlamanız gerekir.

Uygulamanızın ilgilendiği kadarıyla, kuyruk yöneticisi iki rol gerçekleştirir: bir kaynak yöneticisi (kaynakların kuyruklarda ileti olduğu) ve genel iş birimine ilişkin hareket yöneticisi.

Sağlanan örnek programlarla başlayın ve bu programlarda yapılmakta olan çeşitli WebSphere MQ ve veritabanı API çağrıları aracılığıyla çalışın. İlgili API çağrıları, “Örnek WebSphere MQ programları” sayfa 92, MQI ' da kullanılan veri tiplerive (veritabanının kendi API 'si durumunda) veritabanının kendi belgelerini tam olarak belgelemektedir.

Sistemin sınanması
Uygulamanızın ve sisteminizin yalnızca test sırasında çalıştırılarak doğru bir şekilde yapılandırılıp yapılandırılmadığını bilirsiniz. Sağlanan örnek programlardan birini oluşturarak ve çalıştırarak, sistemin yapılandırmasını (kuyruk yöneticisi ile veritabanı arasındaki başarılı iletişimi) sınabilirsiniz.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Db2yapılandırılıyor
DB2 desteği ve yapılandırma bilgileri.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Desteklenen Db2 düzeyleri, IBM WebSphere MQ ayrıntılı sistem gereksinimleri sayfasında tanımlanır.

Not: 32-bit instances of Db2 are not supported on platforms where the queue manager is 64-bit.

Aşağıdakileri yapın:

1. Ortam değişkeni ayarlarını denetleyin.
2. Db2 anahtar yükleme dosyasını yaratın.
3. Kaynak yöneticisi yapılanış bilgileri eklemenizi sağlar.
4. Gerekiyorsa, Db2 yapılanış değiştirgelerini değiştirin.

Bu bilgileri, “Sisteminizi veritabanı koordinasyonu için yapılandırma” sayfa 45içinde sağlanan genel bilgilerle birlikte okuyun.

Uyarı: If you run db2`profile` on UNIX and Linux platforms, the environment variable LIBPATH and LD_LIBRARY_PATH are set. Bu ortam değişkenlerinin `unset` ' e uygun olması önerilir, bkz. *Kısa Kullanım* Kılavuzu.

Db2 ortam değişkeni ayarlarının denetlenmesi

Ensure that your Db2 environment variables are set for queue manager processes ***şu anda olduğu gibi*** your application processes. In particular, you must always set the DB2INSTANCE environment variable ***önce*** you start the queue manager. DB2INSTANCE ortam değişkeni, güncellenmekte olan Db2 veritabanlarının bulunduğu Db2 yönetim ortamını tanıtır. Örneğin:

- UNIX and Linux sistemlerinde şunu kullanın:

```
export DB2INSTANCE=db2inst1
```

- Windows sistemlerinde şunu kullanın:

```
set DB2INSTANCE=DB2
```

On Windows with a Db2 database, you must add the user MUSR_MQADMIN to the DB2USERS group, to enable the queue manager to start.

Db2 anahtar yükleme dosyasının oluşturulması

The easiest way to create the Db2 switch load file is to use the sample file xaswit.mak, which WebSphere MQ provides to build the switch load files for a variety of database products.

On Pencereler systems, you can find xaswit.mak in the directory *MQ_INSTALLATION_PATH*`\tools\c\samples\xatm`. *MQ_INSTALLATION_PATH* WebSphere MQ ' un

kurulu olduğu üst düzey dizini temsil eder. Microsoft Visual C + + ile Db2 anahtar yükleme dosyasını yaratmak için aşağıdaki komutu kullanın:

```
nmake /f xaswit.mak db2swit.dll
```

Oluşturulan anahtar dosyası `c:\Program Files\IBM\WebSphere MQ\exits`içine yerleştirilir.

You can find xaswit.mak in the directory *MQ_INSTALLATION_PATH*`/samp/xatm`. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

xaswit.mak dosyasını, kullanmakta olduğunuz Db2 sürümüne uygun olan satırları *açıklama satırı kaldırma* olarak düzenleyin. Daha sonra, komutu kullanarak makefile komutunu yürütün:

```
make -f xaswit.mak db2swit
```

Oluşturulan 32 bit anahtar yükleme dosyası `/var/mqm/exits`içine yerleştirilir.

Oluşturulan 64 bit anahtar yükleme dosyası `/var/mqm/exits64`içine yerleştirilir.

Db2için kaynak yöneticisi yapılanış bilgileri ekleniyor

Genel iş birimlerinde bir katılımcı olarak Db2 ' yi bir katılımcı olarak bildirmek için kuyruk yöneticisine ilişkin yapılandırma bilgilerini değiştirmeniz gerekir. Yapılandırma bilgilerinin bu şekilde değiştirilmesi, “Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi” sayfa 46ile ilgili daha ayrıntılı bir şekilde açıklanmaktadır.

- Windows ve Linux (x86 ve x86-64 platformları) sistemlerinde, WebSphere MQ Explorer olanağını kullanın. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin.
- Diğer tüm sistemlerde, kuyruk yöneticisinin qm.ini dosyasında yer alan XAResourceManager stanza içindeki anahtar yükleme dosyasının ayrıntılarını belirtin.

Şekil 9 sayfa 50 , eşgüdülenecek veritabanının `mydbname`olarak adlandırıldığı bir XAResourceManager girdisini gösteren bir UNIX örneğidir; bu ad `XAOpenString`' da belirtilmektedir:

```
XAResourceManager:
  Name=mydb2
  SwitchFile=db2swit
  XAOpenString=mydbname,myuser,mypasswd,toc=t
  ThreadOfControl=THREAD
```

Şekil 9. Sample XAResourceManager entry for Db2 on UNIX platforms

Not:

1. `ThreadOfControl=THREAD` , sürüm 8 öncesi Db2 sürümleriyle kullanılamaz. `ThreadOfControl` ve XAOpenString parametresini `toc` aşağıdaki birleşimlerden birine ayarlayın:
   - `ThreadOfControl=THREAD` ve `toc=t`
   - `ThreadOfControl=PROCESS` ve `toc=p`

   JDBC/JTA koordinasyonunu etkinleştirmek için jdbcdb2 XA anahtar yükleme dosyasını kullanıyorsanız, `ThreadOfControl=PROCESS` ve `toc=`pseçeneklerini kullanmanız gerekir.

Db2 yapılanış değiştirgelerinin değiştirilmesi

Kuyruk yöneticisinin eşgüdümleme yaptığı her Db2 veritabanı için, veritabanı ayrıcalıklarını ayarlamalı, tp_mon_name değiştirgesini değiştirmeli ve maxappls değiştirgesini sıfırlamalısınız. Bunu yapmak için aşağıdaki adımları gerçekleştirin:

Veritabanı ayrıcalıklarını ayarla

Kuyruk yöneticisi işlemleri, UNIX and Linux sistemlerinde etkin kullanıcı ve grup mqm ile çalıştırılıyor. Windows sistemlerinde, kuyruk yöneticisini başlatan kullanıcı olarak çalıştırırlar. Bu, aşağıdakilerden biri olabilir:

1. `strmqm` komutunu veren kullanıcı ya da
2. IBM MQSeries Service COM sunucusunun çalıştığı kullanıcı

Varsayılan olarak, bu kullanıcıya MUSR_MQADMIN adı verilir.

Xa_open dizgisinde bir kullanıcı adı ve parola belirtmediyseniz, xa_open çağrısının kimliğini doğrulamak için **kuyruk yöneticisinin altında çalıştığı kullanıcı** Db2 tarafından kullanılır. Bu kullanıcı (örneğin, UNIX and Linux sistemlerindeki kullanıcı mqm) veritabanında en alt düzeyde ayrıcalıklara sahip değilse, veritabanı xa_open çağrısının kimliğini doğrulamayı reddediyor.

Aynı noktalar, uygulama süreciniz için de geçerlidir. xa_open dizgisinde bir kullanıcı adı ve parola belirtmediyseniz, uygulamanızın çalıştığı kullanıcı, Db2 tarafından, ilk MQBEGIN sırasında yapılan xa_open çağrısının kimliğini doğrulamak için kullanılır. Yine, bu kullanıcının çalışabilmek için veritabanında en alt düzeyde ayrıcalıkları olmalıdır.

Örneğin, aşağıdaki Db2 komutlarını vererek, mydbname veritabanında mqm kullanıcı bağlanma yetkisi verin:

```
db2 connect to mydbname
db2 grant connect on database to user mqm
```

Güvenliğe ilişkin daha fazla bilgi için bkz. “Güvenlikle ilgili önemli noktalar” sayfa 59 .

Windows **TP_MON_NAME parametresini değiştirin**

For Db2 for Pencereler systems only, change the TP_MON_NAME configuration parameter to name the DLL that Db2 uses to call the queue manager for dynamic registration.

Use the command `db2 update dbm cfg using TP_MON_NAME mqmax` to name MQMAX.DLL as the library that Db2 uses to call the queue manager. Bu, PATH içinde bir dizinde var olmalıdır.

maxappls parametresini ilk durumuna getir

Bir veritabanına bağlanabilecek uygulama sayısı üst sınırını sınırlayan *maxappls* parametresine ilişkin ayarınızı gözden geçirmeniz gerekebilir. Şu konuya bakın: “Veritabanı ürününün kurulması ve yapılandırılması” sayfa 45.

Oracle' ın yapılandırılması

Oracle desteği ve yapılandırma bilgileri.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki adımları izleyin:

1. Ortam değişkeni ayarlarını denetleyin.
2. Oracle anahtar yükleme dosyasını oluşturun.
3. Kaynak yöneticisi yapılanış bilgileri eklemenizi sağlar.
4. Gerekiyorsa, Oracle yapılanış değiştirgelerini değiştirin.

A current list of levels of Oracle supported by IBM WebSphere MQ is provided at the IBM WebSphere MQ ayrıntılı sistem gereksinimleri page.

Oracle ortam değişkeni ayarlarının denetlenmesi

Ensure that your Oracle environment variables are set for queue manager processes as well as in your application processes. Kuyruk yöneticisini başlatmadan önce, özellikle aşağıdaki ortam değişkenlerini her zaman ayarlayın:

ORACLE_HOME
Oracle ana dizini. Örneğin, UNIX and Linux sistemlerinde şunu kullanın:

```
export ORACLE_HOME=/opt/oracle/product/8.1.6
```

Windows sistemlerinde şunu kullanın:

```
set ORACLE_HOME=c:\oracle\ora81
```

ORACLE_SI
Kullanılmakta olan Oracle SID 'si. İstemci/sunucu bağlanırlığı için Net8 kullanıyorsanız, bu ortam değişkenini ayarlamak zorunda kalmayabilirsiniz. Oracle belgelerinize bakın.

Sonraki örnek, UNIX and Linux sistemlerinde bu ortam değişkeninin ayarlanmasını gösteren bir örnektir:

```
export ORACLE_SID=sid1
```

Windows sistemlerine eşdeğer bir değer:

```
set ORACLE_SID=sid1
```

Not: PATH ortam değişkeni, binaries dizinini (örneğin, `ORACLE_INSTALL_DIR/VERSION/32BIT_NAME/bin` ya da `ORACLE_INSTALL_DIR/VERSION/64BIT_NAME/bin`) içerecek şekilde ayarlanmalıdır; tersi durumda, makineden bağımsız kitaplıkların eksik olduğunu bildiren bir ileti görebilirsiniz.

If you run queue managers on Windows 64 bit systems, then both 64 bit and 32 bit Oracle clients must be installed. Kuyruk yöneticisi 32 bit anahtar yükleme dosyası kullanan 32 bit işlem olarak çalıştığı için her iki istemciyi de kurmanız gerekir. Bu işlem, 32 bit Oracle istemcisi dll dosyasını başlatır.

64 bit kuyruk yöneticisi tarafından yüklenen anahtar yükleme dosyası, Oracle 64 bit istemci kitaplıklarına erişmelidir. IBM WebSphere MQ bir Windows 64 bit sisteminde çalışırken, 32 bit kuyruk yöneticisi 32 bit Oracle istemcisine erişmelidir.

Oracle anahtar yükleme dosyasının oluşturulması

To create the Oracle switch load file, use the sample file `xaswit.mak`, which IBM WebSphere MQ provides to build the switch load files for various database products. On Windows systems, you can find `xaswit.mak` in the directory `C:\Program Files\IBM\WebSphere MQ\tools\c\samples\xatm`. Oracle anahtar yükleme dosyasını Microsoft Visual C + + ile yaratmak için şunu kullanın: `nmake /f xaswit.mak oraswit.dll`

Oluşturulan anahtar dosyası *MQ_INSTALLATION_PATH*`\exit`siçine yerleştirilir.*MQ_INSTALLATION_PATH* IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

You can find `xaswit.mak` in the directory *MQ_INSTALLATION_PATH*`/samp/xatm`. *MQ_INSTALLATION_PATH* IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

Kullanmakta olduğunuz Oracle sürümüne uygun olan hatların açıklamasını kaldırmak için `xaswit.mak` dosyasını düzenleyin. Daha sonra, komutu kullanarak makefile komutunu yürütün:

```
make -f xaswit.mak oraswit
```

Oluşturulan 32 bit anahtar yükleme dosyası `/var/mqm/exits`içine yerleştirilir.

Oluşturulan 64 bit anahtar yükleme dosyası `/var/mqm/exits64`içine yerleştirilir.

Oracleiçin kaynak yöneticisi yapılanış bilgileri ekleniyor

You must modify the configuration information for the queue manager to declare Oracle as a participant in global units of work. Bu şekilde kuyruk yöneticisine ilişkin yapılandırma bilgilerinin değiştirilmesi, “Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi” sayfa 46içinde daha ayrıntılı bir şekilde açıklanmaktadır.

- Windows ve Linux (x86 ve x86-64 platformları) sistemlerinde IBM WebSphere MQ Explorer' u kullanın. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin.
- Diğer tüm sistemlerde, kuyruk yöneticisinin `qm.ini` dosyasındaki XAResourceManager stanza içindeki anahtar yükleme dosyasının ayrıntılarını belirtin.

Şekil 10 sayfa 53 , XAResourceManager girdisini gösteren bir UNIX and Linux sistemleri örneğidir. Tüm hata ve izleme bilgilerinin aynı yere kaydedilebilmesi için, XA açık dizgisine bir `LogDir` (sayfa adı) eklemelisiniz.

```
XAResourceManager:
  Name=myoracle
  SwitchFile=oraswit
  XAOpenString=Oracle_XA+Acc=P/myuser/mypasswd+SesTm=35+LogDir=/tmp+threads=true
  ThreadOfControl=THREAD
```

Şekil 10. Sample XAResourceManager entry for Oracle on UNIX and Linux platforms

Not:

1. Şekil 10 sayfa 53' ta, xa_open dizgisi dört parametre ile birlikte kullanıldı. Ek parametreler, Oracle' ın belgelerinde açıklandığı şekilde eklenebilir.
2. When using the IBM WebSphere MQ parameter `ThreadOfControl=THREAD` you must use the Oracle parameter `+threads=true` in the XAResourceManager stanza.

Xa_open dizesiyle ilgili ek bilgi için *Oracle8 Server Application Developer's Guide* adlı yayına bakın.

Oracle yapılanış değiştirgelerinin değiştirilmesi

Kuyruk yöneticisinin eşgüdümleme yaptığı her Oracle veritabanı için, oturum sayısı üst sınırını gözden geçirmeniz ve veritabanı ayrıcalıklarınızı ayarlamanız gerekir. Bunu yapmak için şu adımları tamamlayın:

Oturum sayısı üst sınırını gözden geçirin
Kuyruk yöneticisine ait süreçlerin gerektirdiği ek bağlantıları dikkate almak için LICENSE_MAX_SEANSS ve PROCESSES ayarlarınızı gözden geçirmeniz gerekebilir. Daha fazla ayrıntı için bkz. “Veritabanı ürününün kurulması ve yapılandırılması” sayfa 45 .

Veritabanı ayrıcalıklarını ayarla
xa_open dizgisinde belirtilen Oracle kullanıcı adının, Oracle belgelerinde açıklandığı gibi, DBA_PENDING_TRANSATIONS görünümüne erişmek için gereken ayrıcalıkları olmalıdır.

Aşağıdaki örnek komutu kullanarak gerekli ayrıcalığa sahip olabilirsiniz:

```
grant select on DBA_PENDING_TRANSACTIONS to myuser;
```

Informixyapılandırılıyor

Informix desteği ve yapılandırma bilgileri.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki adımları izleyin:

1. Uygun Informix Client SDK ' yı kurduğunuzdan emin olun:
   - 32 bit kuyruk yöneticisi ve uygulamaları için 32 bit Informix istemcisi SDK gerekir.
   - 64 bit kuyruk yöneticisi ve uygulamaları 64 bit Informix istemcisi SDK gerektirir.
2. Informix veritabanlarının tarafından doğru bir şekilde oluşturulduğundan emin olun.
3. Ortam değişkeni ayarlarını denetleyin.
4. Informix anahtar yükleme dosyasını oluşturun.
5. Kaynak yöneticisi yapılanış bilgileri eklemenizi sağlar.

A current list of levels of Informix supported by WebSphere MQ is provided at theIBM WebSphere MQ ayrıntılı sistem gereksinimleri page.

Informix veritabanlarının doğru bir şekilde yaratıldığından emin olma

Bir WebSphere MQ kuyruk yöneticisi tarafından eşgüdümlenecek her Informix veritabanı, `log` değiştirgesini belirterek yaratılmalıdır. Örneğin:

```
create database mydbname with log;
```

WebSphere MQ kuyruk yöneticileri, yaratma sırasında belirtilen `log` parametresine sahip olmayan Informix veritabanlarını koordine edemiyorlar. If a queue manager attempts to coordinate an Informix database that does not have the `log` parameter specified on creation, the xa_open call to Informix fails, and a number of FFT errors are generated.

Informix ortam değişkeni ayarlarının denetlenmesi

Ensure that your Informix environment variables are set for queue manager processes ***şu anda olduğu gibi*** your application processes. Kuyruk yöneticisini başlatmadan **önce** , aşağıdaki ortam değişkenlerini her zamansetöncesetayarlayın:

INFORMIXDIR
Informix ürün kuruluşunun dizini.

- 32 bit UNIX and Linux uygulamaları için aşağıdaki komutu kullanın:

  ```
  export INFORMIXDIR=/opt/informix/32-bit
  ```

- 64 bit UNIX and Linux uygulamaları için aşağıdaki komutu kullanın:

  ```
  export INFORMIXDIR=/opt/informix/64-bit
  ```

- Windows uygulamaları için aşağıdaki komutu kullanın:

  ```
  set INFORMIXDIR=c:\informix
  ```

Hem 32 bit, hem de 64 bit uygulamaları desteklemesi gereken 64 bit kuyruk yöneticisi olan sistemlerde, hem Informix 32 bit hem de 64 bit istemci SDK'lerine gereksinim duyarsınız. Anahtar yükleme dosyası yaratmak için kullanılan örnek makefile `xaswit.mak`, hem ürün kuruluş dizinlerini de ayarlar.

INFORMIXSERVER
Informix sunucusunun adı. Örneğin, UNIX and Linux sistemlerinde şunu kullanın:

```
export INFORMIXSERVER=hostname_1
```

Windows sistemlerinde şunu kullanın:

```
set INFORMIXSERVER=hostname_1
```

ONCONFIG
Informix sunucusu yapılandırma dosyasının adı. Örneğin, UNIX and Linux sistemlerinde şunu kullanın:

```
export ONCONFIG=onconfig.hostname_1
```

Windows sistemlerinde şunu kullanın:

```
set ONCONFIG=onconfig.hostname_1
```

Informix anahtar yükleme dosyasının oluşturulması

Informix anahtar yükleme dosyasını oluşturmak için xaswit.makörnek dosyasını kullanın; WebSphere MQ , çeşitli veritabanı ürünlerine ilişkin anahtar yükleme dosyalarını oluşturmak için sağlar. On Pencereler systems, you can find xaswit.mak in the directory *MQ_INSTALLATION_PATH*`\tools\c\samples\xatm`. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder. Informix anahtar yükleme dosyasını Microsoft Visual C + + ile yaratmak için şunu kullanın:

```
nmake /f xaswit.mak infswit.dll
```

Oluşturulan anahtar dosyası `c:\Program Files\IBM\WebSphere MQ\exits`içine yerleştirilir.

You can find xaswit.mak in the directory *MQ_INSTALLATION_PATH*`/samp/xatm`. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

xaswit.mak dosyasını, kullanmakta olduğunuz Informix sürümüne uygun olan satırları *açıklama satırı kaldırma* olarak düzenleyin. Daha sonra, komutu kullanarak makefile komutunu yürütün:

```
make -f xaswit.mak infswit
```

Oluşturulan 32 bit anahtar yükleme dosyası `/var/mqm/exits`içine yerleştirilir.

Oluşturulan 64 bit anahtar yükleme dosyası `/var/mqm/exits64`içine yerleştirilir.

Informixiçin kaynak yöneticisi yapılandırma bilgileri ekleniyor

You must modify the configuration information for the queue manager to declare Informix as a participant in global units of work. Bu şekilde kuyruk yöneticisine ilişkin yapılandırma bilgilerinin değiştirilmesi, “Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi” sayfa 46' ta daha ayrıntılı bir şekilde açıklanmaktadır.

- Windows ve Linux (x86 ve x86-64 platformları) sistemlerinde, WebSphere MQ Explorer olanağını kullanın. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin.
- Diğer tüm sistemlerde, kuyruk yöneticisinin qm.ini dosyasındaki XAResourceManager stanza içindeki anahtar yükleme dosyasının ayrıntılarını belirtin.

Şekil 11 sayfa 56 , eşgüdülenecek veritabanının `mydbname`olarak adlandırıldığı bir qm.ini XAResourceManager girdisini gösteren bir UNIX örneğidir; bu ad `XAOpenString`' da belirtilmektedir:

```
XAResourceManager:
  Name=myinformix
  SwitchFile=infswit
  XAOpenString=DB=mydbname@myinformixserver\;USER=myuser\;PASSWD=mypasswd
  ThreadOfControl=THREAD
```

Şekil 11. Sample XAResourceManager entry for Informix on UNIX platforms

Not: By default the sample xaswit.mak on UNIX platforms creates a switch load file that uses threaded Informix libraries. Bu Informix kitaplıklarını kullanırken ThreadOf(Threadof) denetiminin THREAD (THREAD) olarak ayarlandığından emin olmalısınız. Şekil 11 sayfa 56' ta qm.ini dosyası XAResourceManager stanza özniteliği ThreadOfDenetimi THREAD olarak ayarlıdır. THEAD belirtildiğinde, uygulamalar iş parçacıklı Informix kitaplıkları ve WebSphere MQ yivli API kitaplıkları kullanılarak oluşturulmalıdır.

XAOpenString özniteliği veritabanı adını içermelidir, ardından @ simgesiyle ve ardından Informix sunucusu adı izler.

Yivsiz Informix kitaplıklarını kullanmak için, qm.ini dosyasının XAResourceManager stanza özniteliğinin ThreadOfDenetimi 'nin PROCESS olarak ayarlandığından emin olmanız gerekir. Ayrıca, örnek xaswit.mak:' da aşağıdaki değişiklikleri de yapmanız gerekir:

1. Yivsiz bir anahtar yükleme dosyasının neslinden açıklama satırı kaldırın.
2. Yivli anahtar yükleme dosyasının neslinden açıklama yapın.

Sybase yapılanışı

Sybase desteği ve yapılandırma bilgileri.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki adımları izleyin:

1. Ensure you have installed the Sybase XA libraries, for example by installing the XA DTM option.
2. Ortam değişkeni ayarlarını denetleyin.
3. Sybase XA desteğini etkinleştirin.
4. Sybase anahtar yükleme dosyasını yaratın.
5. Kaynak yöneticisi yapılanış bilgileri eklemenizi sağlar.

A current list of levels of Sybase supported by WebSphere MQ is provided at the IBM WebSphere MQ ayrıntılı sistem gereksinimleri page.

Sybase ortam değişkeni ayarlarının denetlenmesi

Ensure that your Sybase environment variables are set for queue manager processes ***şu anda olduğu gibi*** your application processes. Kuyruk yöneticisini başlatmadan **önce** , aşağıdaki ortam değişkenlerini her zamansetöncesetayarlayın:

Sybase
Sybase ürün kuruluşunun konumu. Örneğin, UNIX and Linux sistemlerinde şunu kullanın:

```
export SYBASE=/sybase
```

Windows sistemlerinde şunu kullanın:

```
set SYBASE=c:\sybase
```

SYBASE_OCS
Sybase istemci dosyalarını yüklediğiniz SYBASE altındaki dizin. Örneğin, UNIX and Linux sistemlerinde şunu kullanın:

```
export SYBASE_OCS=OCS-12_0
```

Windows sistemlerinde şunu kullanın:

```
set SYBASE_OCS=OCS-12_0
```

Sybase XA desteğini etkinleştirme

Within the Sybase XA configuration file $SYBASE/$SYBASE_OCS/xa_config, define a Logical Resource Manager (LRM) for each connection to the Sybase server that is being updated. $SYBASE/$SYBASE_OCS/xa_config içeriğine ilişkin bir örnek, Şekil 12 sayfa 57içinde gösterilmektedir.

```
# The first line must always be a comment
[xa]
  LRM=lrmname
  server=servername
```

Şekil 12. $SYBASE/ $SYBASE_OCS/xa_config dosyasının örnek içeriği

Sybase anahtar yükleme dosyasının oluşturulması

Sybase anahtar yükleme dosyasını yaratmak için, WebSphere MQile sağlanan örnek dosyaları kullanın. Windows sistemlerinde, C:\Program Files\IBM\WebSphere MQ\tools\c\samples\xatmdizininde xaswit.mak öğesini bulabilirsiniz. Sybase anahtar yükleme dosyasını Microsoft Visual C + + ile yaratmak için şunu kullanın:

```
nmake /f xaswit.mak sybswit.dll
```

Oluşturulan anahtar dosyası c:\Program Files\IBM\WebSphere MQ\exitsiçine yerleştirilir.

You can find xaswit.mak in the directory *MQ_INSTALLATION_PATH*/samp/xatm. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

xaswit.mak dosyasını, kullanmakta olduğunuz Sybase sürümüne uygun olan satırları *açıklama satırı kaldırma* olarak düzenleyin. Daha sonra, komutu kullanarak makefile komutunu yürütün:

```
make -f xaswit.mak sybswit
```

Oluşturulan 32 bit anahtar yükleme dosyası /var/mqm/exitsiçine yerleştirilir.

Oluşturulan 64 bit anahtar yükleme dosyası /var/mqm/exits64içine yerleştirilir.

Sybaseiçin kaynak yöneticisi yapılanış bilgileri ekleniyor

You must modify the configuration information for the queue manager to declare Sybase as a participant in global units of work. Yapılandırma bilgilerinin değiştirilmesi, “Yapılanış bilgilerinin kuyruk yöneticisine eklenmesi” sayfa 46içinde daha ayrıntılı bir şekilde açıklanmaktadır.

- Windows ve Linux (x86 ve x86-64 platformları) sistemlerinde, WebSphere MQ Explorer olanağını kullanın. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin.

- Diğer tüm sistemlerde, kuyruk yöneticisinin qm.ini dosyasında yer alan XAResourceManager stanza içindeki anahtar yükleme dosyasının ayrıntılarını belirtin.

Şekil 13 sayfa 58 , Sybase XA yapılandırma dosyasında ($SYBASE/$SYBASE_OCS/xa_config) *lrmname* LRM tanımlamasıyla ilişkili veritabanını kullanan bir UNIX and Linux örneğini gösterir. XA işlev çağrılarının günlüğe kaydedilmesini istiyorsanız bir günlük dosyası adı ekleyin:

```
XAResourceManager:
  Name=mysybase
  SwitchFile=sybswit
  XAOpenString=-Uuser -Ppassword -Nlrmname -L/tmp/sybase.log -Txa
  ThreadOfControl=THREAD
```

Şekil 13. Sample XAResourceManager entry for Sybase on UNIX and Linux platforms

Sybaseile çok iş parçacıklı programlar kullanılması

If you are using multi-threaded programs with WebSphere MQ global units of work incorporating updates to Sybase, you ***gerekir*** use the value THREAD for the ThreadOfControl parameter. Ayrıca, programınızı (ve anahtar yükleme dosyasını) iş parçacığı korumalı Sybase kitaplıklarıyla (_r sürümleri) bağlayıp bağladığınızdan emin olun. ThreadOfDenetim parametresi için THREAD değerinin kullanılması, Şekil 13 sayfa 58içinde gösterilir.

Birden çok veritabanı yapılandırması

Kuyruk yöneticisini yapılandırmak istiyorsanız, genel iş birimleri içinde birden çok veritabanına yapılan güncellemeler içerilebilmesi için, her veritabanı için bir XAResourceManager kısmı ekleyin.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Veritabanlarının tümü aynı veritabanı yöneticisi tarafından yönetiliyorsa, her bir tanza ayrı bir veritabanı tanımlar. Her bir stanza aynı *SwitchFile*değerini belirtir, ancak güncellenmekte olan veritabanının adını belirttiğinden, *XAOpenString* içeriğinin içeriği farklıdır. Örneğin, Şekil 14 sayfa 58 içinde gösterilen stanzas, kuyruk yöneticisini UNIX and Linux sistemlerinde *MQBankDB* ve *MQFeeDB* Db2 veritabanlarıyla birlikte yapılandırır.

Önemli: Aynı veritabanını gösteren birden çok stanzada bulunamaz. Bu yapılandırma, hiçbir koşulda çalışmaz ve bu yapılandırmayı dendiğinizde başarısız olur.

You will receive errors of the form `when the MQ code makes its second xa_open call in any process in this environment, the database software fails the second xa_open with a -5 error, XAER_INVAL`.

```
XAResourceManager:
  Name=DB2 MQBankDB
  SwitchFile=db2swit
  XAOpenString=MQBankDB

XAResourceManager:
  Name=DB2 MQFeeDB
  SwitchFile=db2swit
  XAOpenString=MQFeeDB
```

Şekil 14. Birden çok Db2 veritabanı için örnek XAResourceManager girdileri

Güncellenecek veritabanları farklı veritabanı yöneticileri tarafından yönetiliyorsa, her biri için bir XAResourceManager stanza ekleyin. Bu durumda, her bir stanza farklı bir *SwitchFile*(SwitchFile) belirtir.

Örneğin, *MQFeeDB* , Oracle tarafından DB2yerine yönetiliyorsa, UNIX and Linux sistemlerinde aşağıdaki stanzaları kullanın:

```
XAResourceManager:
  Name=DB2 MQBankDB
  SwitchFile=db2swit
  XAOpenString=MQBankDB

XAResourceManager:
  Name=Oracle MQFeeDB
  SwitchFile=oraswit
  XAOpenString=Oracle_XA+Acc=P/myuser/mypassword+SesTm=35+LogDir=/tmp/ora.log+DB=MQFeeDB
```

Şekil 15. Bir DB2 ve Oracle veritabanı için örnek XAResourceManager girişleri

Prensipte, tek bir kuyruk yöneticisiyle yapılandırılabilecek veritabanı örneği sayısı için bir sınır yoktur.

Not: Genel iş birimleri içinde birden çok veritabanı güncellemesinde Informix veritabanlarının da eklenmesine ilişkin bilgi edinmek için, ürünün benioku dosyasına bakın.

Güvenlikle ilgili önemli noktalar

Veritabanınızı XA modeli altında çalıştırmanın dikkat edilmesi gereken noktalar.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki bilgiler yalnızca kılavuzluk için sağlanmıştır. Tüm durumlarda, veritabanını XA modeli altında çalıştırmanın güvenlik etkilerini saptamak için, veritabanı yöneticisiyle birlikte sağlanan belgelere bakın.

Bir uygulama işlemi, `MQBEGIN` komutunu kullanarak genel bir çalışma biriminin başlangıcını belirtir. Bir uygulama sorunu olan ilk `MQBEGIN` çağrısı, istemci kitaplık kodunu xa_open giriş noktasında çağırarak tüm katılımcı veritabanlarına bağlanır. Tüm veritabanı yöneticileri, XAOpenString' de bir kullanıcı kimliği ve parola sağlamak için bir mekanizma sağlar. Bu, kimlik doğrulama bilgileri akışının yegane zamanidir.

UNIX and Linux altyapılarında, fastpath uygulamalarının MQI çağrıları yaparken mqm etkin bir kullanıcı kimliğiyle çalışması gerektiğini unutmayın.

Kişi XA kaynak yöneticisiyle kaybolduğunda dikkat edilecek noktalar

Kuyruk yöneticisi, veritabanı yöneticilerine kabul edilmemesine tahammül eder. Bu, kuyruk yöneticisini veritabanı sunucusundan bağımsız olarak başlatabileceğiniz ve durdurabileceğiniz anlamına gelir. Kişi geri yüklendiğinde, kuyruk yöneticisi ve veritabanı yeniden uyumlulaştırır. Belirsiz iş birimlerini el ile çözmek için rsvmqtrn komutunu da kullanabilirsiniz.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Olağan işlemlerde, yapılandırma adımlarını tamamladıktan sonra yalnızca en az miktarda yönetim gereklidir. Kuyruk yöneticisi veritabanı yöneticilerinin kullanılabilir olmamasını kabul etmesinden dolayı, denetim işi kolaylaşılır. Özellikle şu anlama geliyor:

- Kuyruk yöneticisi, her veritabanı yöneticisini başlatmadan önce herhangi bir zamanda başlayabilir.
- Veritabanı yöneticilerinden biri kullanılamaz duruma geldiyse, kuyruk yöneticisinin durdurup yeniden başlatılması gerekmez.

Bu, kuyruk yöneticisini veritabanı sunucusundan bağımsız olarak başlatmanızı ve durdurmanızı sağlar.

Kuyruk yöneticisi ve veritabanı arasında bağlantı kaybolduğunda, her ikisi de yeniden kullanılabilir duruma geldiğinde yeniden eşzamanlamak gerekir. Yeniden eşzamanlama, bu veritabanıyla ilgili herhangi bir belirsiz iş biriminin tamamlandığı işlemdir. Genel olarak, bu durum kullanıcı müdahalelerine gerek kalmadan otomatik olarak gerçekleşir. Kuyruk yöneticisi, veritabanından kuşku içinde olan bir iş birimi

listesi için veritabanı ister. Daha sonra, veritabanına bu belirsiz iş birimlerinin her birini kesinleştirmesi ya da geri alma işlemi için talimat verir.

Kuyruk yöneticisi başlatıldığında, her bir veritabanıyla yeniden eşitlenir. Tek bir veritabanı kullanılamaz hale geldiğinde, kuyruk yöneticisi tarafından yeniden kullanılabilir duruma getirildiğinde, veritabanının sonraki yeniden eşzamanlanması için yalnızca o veritabanının yeniden eşzamanlı kılınması gerekir.

Yeni genel iş birimleri MQBEGIN ile başlatıldığı için, kuyruk yöneticisi daha önce kullanılamayan bir veritabanıyla otomatik olarak bağlantı kazanır. Bunu, veritabanı istemcisi kitaplığındaki xa_open işlevini çağırarak yapar. Bu xa_open çağrısı başarısız olursa, MQBEGIN, MQCC_UYARI tamamlama koduyla ve MQRC_PARTICIPANT_NOT_AVAILATION bir neden kodunu döndürür. MQBEGIN çağrısını daha sonra yeniden deneyebilirsiniz.

MQBEGIN sırasında başarısız olan bir veritabanına ilişkin güncellemeleri içeren genel bir iş birimi girişiminde bulunmaya devam etme. Bu veritabanıyla ilgili güncellemelerin yapılabileceği bir bağlantı olmayacak. Tek seçenekleriniz programı sona erdirmek ya da veritabanının yeniden kullanılabilir duruma gelebileceği umuduyla MQBEGIN düzenli olarak yeniden denenmesini sağlar.

Diğer bir seçenek olarak, `rsvmqtrn` komutunu kullanarak, açık bir şekilde tüm belirsiz iş birimlerini çözümleyebilirsiniz.

Belirsiz iş birimleri

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Veritabanı yöneticisinin hazırlanması talimatı verildikten sonra kuyruk yöneticisiyle iletişim kesilirse, veritabanı belirsiz iş birimleriyle birlikte bırakılabilir. Veritabanı sunucusu, sonucu kuyruk yöneticisinden (kesinleştirme ya da geri alma) aldıktan sonra, güncellemelerle ilişkili veritabanı kilitlerini alıkoyması gerekir.

Bu kilitler, diğer uygulamaların veritabanı kayıtlarını güncellemesini ya da okumasını önlediğinden, yeniden eşzamanlama işlemi mümkün olan en kısa zamanda gerçekleşmeli.

Bir nedenden dolayı, kuyruk yöneticisinin veritabanıyla otomatik olarak yeniden eşzamanlanması için bekleyemezseniz, veritabanı güncellemelerini el ile kesinleştirmesi ya da geri almak için veritabanı yöneticisi tarafından sağlanan olanakları kullanabilirsiniz. *X/Open Distributed Transaction Processing: The XA Specification*(X/Open Distributed Transaction Processing: XA Belirtimi) içinde bu, *buluşsal* karar alma Veri bütünlüğünü tehlikeye atmanın olasılığı nedeniyle, bunu yalnızca son çare olarak kullanın; örneğin, diğer tüm katılımcılar güncellemelerini kesinleştirdiğinde yanlışlıkla veritabanı güncellemelerini geriye doğru geri alabilirsiniz.

Kuyruk yöneticisini yeniden başlatmak çok daha iyi olur ya da otomatik yeniden eşzamanlamayı başlatmak için veritabanı yeniden başlatıldığında `rsvmqtrn` komutunu kullanın.

Bekleyen iş birimlerinin dspmqtrn komutu ile görüntülenmesi

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bir veritabanı yöneticisi kullanılamazken, o veritabanını içeren olağanüstü genel iş birimlerinin durumunu denetlemek için **dspmqtrn** komutunu kullanabilirsiniz.

dspmqtrn komutu, yalnızca bir ya da daha çok katılımcının kuşku içinde olduğu iş birimlerini görüntüler. Katılımcılar, hazırlanan güncellemeleri kesinleştirmek ya da geri almak için kuyruk yöneticisinden alınan kararı bekliyor.

Bu genel iş birimlerinin her biri için, her bir katılımcının durumu **dspmqtrn**tarafından çıkışta görüntülenir. İş birimi belirli bir kaynak yöneticisinin kaynaklarını güncellemezse, bu işlem görüntülenmez.

Belirsiz bir iş birimine ilişkin olarak, bir kaynak yöneticisinin aşağıdaki şeylerden birini yapmış olduğu söylenmektedir:

Hazırlandı
Kaynak yöneticisi güncelleştirmelerini kesinleştirmeye hazır.

Kesinleştirildi
Kaynak yöneticisi güncelleştirmelerini kesinleştirdi.

Geri alındı
Kaynak yöneticisi güncelleştirmelerini geriye işledi.

Katılmıştır
Kaynak yöneticisi bir katılımcıdır, ancak güncellemelerini hazırlamamıştır, kesinleştirmez ya da geriye işlemez.

Kuyruk yöneticisi yeniden başlatıldığında, her bir veritabanının belirsiz genel iş birimleri listesi için XAResourceManager kısmına sahip olmasını ister. Veritabanı yeniden başlatılmamışsa ya da başka bir şekilde kullanılamıyorsa, kuyruk yöneticisi henüz bu iş birimleri için son kazanımları veritabanına teslim edemez. Belirsiz iş birimlerinin sonucu, veritabanı yeniden kullanılabilir olduğunda ilk fırsatta veritabanına teslim edilir.

Bu durumda, yeniden eşzamanlama işlemi gerçekleşinceye kadar, veritabanı yöneticisi *hazırlandı* durumunda olduğu gibi raporlanır.

`dspmqtrn` komutu belirsiz bir iş birimi görüntülediğinde, ilk olarak, katılım gösterebilecek tüm kaynak yöneticilerini listeler. Bunlar, durumu belirsiz bir iş birimine göre bildirirken kaynak yöneticilerinin *Ad* yerine kullanılan benzersiz bir tanıtıcı (*RMTnt*) tahsis edilir.

Örnek dspmqtrn çıkışı , aşağıdaki komutun yayınının sonucunu gösterir:

```
dspmqtrn -m MY_QMGR
```

```
AMQ7107: Resource manager 0 is MQSeries.
AMQ7107: Resource manager 1 is DB2 MQBankDB.
AMQ7107: Resource manager 2 is DB2 MQFeeDB.

AMQ7056: Transaction number 0,1.
    XID: formatID 5067085, gtrid_length 12, bqual_length 4
         gtrid [3291A50600002013746573 74]
         bqual [00000001]
AMQ7105: Resource manager 0 has committed.
AMQ7104: Resource manager 1 has prepared.
AMQ7104: Resource manager 2 has prepared.
```

Burada *İşlem numarası* , rsvmqtrn komutuyla birlikte kullanılabilecek işlemin tanıtıcısıdır. AMQ7056 iletiyle ilgili daha fazla bilgi için bkz. AMQ7000-7999: WebSphere MQ ürünü . *XID* değişkenleri, *X/Open XA Specification*' ın bir parçasıdır; bu belirtimle ilgili en güncel bilgiler için şu konuya bakın: https://publications.opengroup.org/c193.

Şekil 16. Örnek dspmqtrn çıkışı

Örnek dspmqtrn çıkışı içindeki çıkış, kuyruk yöneticisiyle ilişkilendirilmiş üç kaynak yöneticisi olduğunu gösterir. İlki, kuyruk yöneticisinin kendisi olan kaynak yöneticisi 0' dir. Diğer iki kaynak yöneticisi eşgörünümü, MQBankDB ve MQFeeDB Db2 veritabanlarıdır.

Bu örnek, yalnızca tek bir belirsiz iş birimi gösterir. Tüm üç kaynak yöneticisi için bir ileti yayınlanır. Bu ileti, iş birimi içindeki kuyruk yöneticisinde ve her iki Db2 veritabanı için güncelleme yapıldığı anlamına gelir.

Kuyruk yöneticisine yapılan güncellemeler, kaynak yöneticisi **0**, *kesinleştirilmiştir*. The updates to the Db2 databases are in *hazırlandı* state, which means that Db2 must have become unavailable before it was called to commit the updates to the *MQBankDB* and *MQFeeDB* databases.

Belirsiz iş birimi, XID (*işlem tanıtıcısı*) adı verilen bir dış tanıtıcıya sahiptir. Bu, genel iş birimi bölümünü tanımlamak için kuyruk yöneticisi tarafından Db2 ' ya verilen bir veri parçasıdır.

rsvmqtrn komutuyla iş bekleyen iş birimlerinin çözümleniyor
Kuyruk yöneticisi ve DB2 yeniden uyumlulaştırıldığında, olağanüstü iş birimleri tamamlandı.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Şekil 16 sayfa 61 içinde gösterilen çıkış, kesinleştirme kararının henüz hem DB2 veritabanlarına teslim edilmemesine ilişkin tek bir belirsiz iş birimi gösterir.

To complete this unit of work, the queue manager and DB2 need to resynchronize when DB2 next becomes available. Kuyruk yöneticisi, yeni iş birimlerinin başlangıcını DB2ile yeniden bağlantı kurmak için bir fırsat olarak kullanır. Diğer bir seçenek olarak, kuyruk yöneticisine **rsvmqtrn** komutunu kullanarak açık bir şekilde yeniden eşzamanlama konusunda bilgi verebiliyorsunuz.

DB2 yeniden başlatıldıktan hemen sonra bu işlemi hemen yapın; böylece, belirsiz çalışma birimiyle ilişkili veritabanı kilitleri olabildiğince çabuk serbest bırakılır. Kuyruk yöneticisini, tüm belirsiz iş birimlerini çözümlemek için bildiren -a seçeneğini kullanın. Aşağıdaki örnekte, DB2 yeniden başlatılırmış, bu nedenle kuyruk yöneticisi belirsiz iş birimini çözebilir:

```
> rsvmqtrn -m MY_QMGR -a
Any in-doubt transactions have been resolved.
```

Karma kazanımlar ve hatalar
Kuyruk yöneticisi iki aşamalı kesinleştirme protokolünü kullansa da, bu durum karma kazanımlar içeren bazı iş birimlerinin olasılıkları tamamen ortadan kaldırılmaz. Bu, bazı katılımcıların güncelleştirmelerini kesinleştirdikleri ve bazılarının güncelleştirmelerini geri gönderdikleri yerdir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Tek bir iş birimi olarak güncellenmesi gereken paylaşılan kaynaklar artık tutarlı durumda olmadığı için, karma bir sonuçla tamamlanan iş birimleri ciddi sonuçlara yol göstermektedir.

Karma kazanımlar daha çok, kuyruk yöneticisinin belirsiz iş birimlerinin çözülmesi için kuyruk yöneticisine izin vermek yerine, iş birimleri hakkında buluşsal kararlar verildiğinde ortaya çıkar. Bu tür kararlar, kuyruk yöneticisinin denetiminden dışındadır.

Kuyruk yöneticisi karma bir sonuç saptadığında, FFST bilgi üretir ve hata günlüklerindeki başarısızlığı iki iletiyle birlikte oluşturur:

- Veritabanı yöneticisi kesinleştirilmek yerine geri dönerse:

```
AMQ7606 A transaction has been committed but one or more resource
        managers have rolled back.
```

- Veritabanı yöneticisi geri dönüş yerine kesinleştirirse:

```
AMQ7607 A transaction has been rolled back but one or more resource
        managers have committed.
```

Daha fazla ileti, buluşsal olarak zarar gören veritabanlarını tanıtır. Bundan sonra, etkilenen veritabanlarında tutarlılığı yerel olarak geri yüklemek sizin sorumluluğunuz. Bu, yanlış kesinleştirilmiş ya da geriye işlenen güncellemeyi yalıtmak için önce gerekli olan, veritabanını el ile geri almak ya da yinelemek için önce gereksinim dumanız gereken karmaşık bir yordamdır.

Yapılandırma bilgilerinin değiştirilmesi
Kuyruk yöneticisi, genel iş birimlerinin koordinasyonu için başarıyla başlatıldıktan sonra, kaynak yöneticisi yapılanış bilgilerinin hiçbirini değiştirmeyin.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Yapılandırma bilgilerini değiştirmeniz gerekirse, istediğiniz zaman yapabilirsiniz, ancak değişiklikler kuyruk yöneticisi yeniden başlatılıncaya kadar yürürlüğe girmez.

Bir veritabanına ilişkin kaynak yöneticisi yapılanış bilgilerini kaldırırsanız, kuyruk yöneticisinin bu veritabanı yöneticisiyle bağlantı kurmasını etkili bir şekilde kaldırabilirsiniz.

Hiçbir zaman , kaynak yöneticisi yapılanış bilgilerinizin herhangi birinde *Ad* özniteliğini değiştirmez. Bu öznitelik, veritabanı yöneticisi yönetim ortamını kuyruk yöneticisine benzersiz olarak tanıtır. Bu benzersiz tanıtıcıyı değiştirirseniz, kuyruk yöneticisi veritabanının kaldırıldığını ve tümüyle yeni bir yönetim ortamı eklendiğini varsayar. Kuyruk yöneticisi hala bekleyen iş birimlerini eski *Ad*ile ilişkilendiriyor, büyük olasılıkla veritabanından belirsiz durumda bırakılıyor.

Veritabanı yöneticisi yönetim ortamları kaldırılıyor
Bir veritabanını kalıcı olarak yapılanışınızdan kaldırmanız gerekiyorsa, kuyruk yöneticisini yeniden başlatmadan önce veritabanının kuşku içinde olmadığından emin olun.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Veritabanı ürünleri, belirsiz hareketleri listelemek için komutlar sağlar. Belirsiz hareketler varsa, önce kuyruk yöneticisinin veritabanıyla yeniden eşzamanlanması için izin verin. Kuyruk yöneticisini başlatarak bunu yapın. You can verify that resynchronization has taken place by using the **rsvmqtrn** command or the database's own command for viewing in-doubt units of work. Yeniden eşzamanlamanın gerçekleştirildiğinden emin olun, kuyruk yöneticisini sona erdirin ve veritabanının yapılandırma bilgilerini kaldırın.

Bu yordamı gözlemleyemezseniz, kuyruk yöneticisi o veritabanıyla ilgili tüm belirsiz iş birimlerini anımsayamaz. Kuyruk yöneticisi her yeniden başlatıldığında bir uyarı iletisi (AMQ7623) yayınlanır. Bu veritabanını kuyruk yöneticisiyle yeniden yapılandıramayacaksa, kuyruk yöneticisine veritabanının katılımını unutmasını bildirmek için, **rsvmqtrn** komutunun -r seçeneğini kullanarak, veritabanının katılımını unutmasını isteyin. Kuyruk yöneticisi, yalnızca tüm katılımcılarla belirsiz hareketler tamamlandığında bu tür işlemleri unutur.

Bazı kaynak yöneticisi yapılanış bilgilerini geçici olarak kaldırmanız gerekebilecek zamanlar vardır. UNIX and Linux sistemlerinde bu, daha sonra kolayca yeniden yürürlüğe konması için stanza hakkında yorum yaparak en iyi şekilde elde edilir. Kuyruk yöneticisi belirli bir veritabanı ya da veritabanı yöneticisine her bağlantısında hata varsa, bunu yapmaya karar verebilirsiniz. İlgili kaynak yöneticisi yapılanış bilgilerinin geçici olarak kaldırılması, kuyruk yöneticisinin diğer tüm katılımcıların yer aldığı genel iş birimlerini başlatmasına olanak sağlar. Aşağıda, açıklamalı bir XA`ResourceManager` stanza örneği verilmiştir:

```
# This database has been temporarily removed
#XAResourceManager:
#  Name=mydb2
#  SwitchFile=db2swit
#  XAOpenString=mydbname,myuser,mypassword,toc=t
#  ThreadOfControl=THREAD
```

Şekil 17. Commented- out XAResourceManager stanza on UNIX and Linux systems

Pencereler sistemlerinde, veritabanı yöneticisi örneğiyle ilgili bilgileri silmek için WebSphere MQ Gezgini 'ni kullanın. *Ad* alanında, adı yeniden yürürlüğe getirirken doğru adı yazmak için büyük özen gösteriniz. If you mistype the name, you may face in-doubt problems, as described in “Yapılandırma bilgilerinin değiştirilmesi” sayfa 62.

XA dinamik kaydı

XA belirtimi, bir hareket yöneticisinin bir kaynak yöneticisine yaptığı `xa_*` çağrılarının sayısını azaltmanın bir yolunu sağlar. Bu eniyileme, *dinamik kayıt*olarak bilinir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Dinamik kayıt, DB2tarafından desteklenir. Diğer veritabanları bunu destekleyebilir; ayrıntılar için veritabanı ürününüze ilişkin belgelere bakın.

dinamik kayıt optimizasyonu neden faydalı? Uygulamanınızda, bazı genel iş birimleri veritabanı tablolarına ilişkin güncellemeler içerebilir; diğerleri bu güncellemeleri içermeyebilir. Bir veritabanının çizelgelerinde kalıcı güncelleme yapılmadığında, o veritabanını MQCMIT sırasında oluşan kesinleştirme protokolünde içermeye gerek yoktur.

Veritabanınızın dinamik kaydı destekleyip desteklemediğini, uygulamanızın bir WebSphere MQ bağlantısı üzerinde ilk MQBEGIN çağrısı sırasında `xa_open` çağrısını çağrılacağını. Ayrıca, sonraki MQDISC çağrısında da `xa_close` çağrısını çağırır. Sonraki XA çağrılarının örüntüleri, veritabanının dinamik kaydı destekleyip desteklememesine bağlıdır:

Veritabanınız dinamik kaydı desteklemiyorsa ...
Every global unit of work involves several XA function calls made by WebSphere MQ code into the database client library, regardless of whether you made a persistent update to the tables of that database within your unit of work. Bu üyeler şunlardır:

- Uygulama sürecindenxa_start ve xa_end . Bunlar, genel bir iş biriminin başlangıcını ve sonunu bildirmek için kullanılır.
- Kuyruk yöneticisi aracısındanxa_prepare, xa_commitve xa_rollback , amqzlaa0. Bunlar, genel çalışma biriminin sonucunu sağlamak için kullanılır: kesinleştirme ya da geriye işleme kararı.

Ayrıca, kuyruk yöneticisi aracısı işlemi de ilk MQBEGIN sırasında xa_open çağrısını da çağırır.

Veritabanınız dinamik kaydı destekliyorsa ...
WebSphere MQ kodu, yalnızca gerekli XA işlev çağrılarını yapar. Veritabanı kaynaklarına yönelik sürekli güncellemeleri **içermeyen** genel iş birimi için, veritabanına XA çağrısı **yok** XA çağrıları vardır. **has** ' in bu tür kalıcı güncellemelere dahil olan genel bir iş birimi için aramalar:

- xa_end from the application process to declare the end of the global unit of work.
- Kuyruk yöneticisi aracısındanxa_prepare, xa_commitve xa_rollback , amqzlaa0. Bunlar, genel çalışma biriminin sonucunu sağlamak için kullanılır: kesinleştirme ya da geriye işleme kararı.

Dinamik kayıt çalışması için, veritabanının geçerli genel iş birimine dahil edilmesi istediği kalıcı bir güncelleme gerçekleştirdiğinde WebSphere MQ ' a bir şekilde söylemesinin bir yolu olması hayati önem taşıyan bir yöntemdir. WebSphere MQ , bu amaç için ax_reg işlevini sağlar.

Veritabanı istemci kodu, uygulama sürecinizde çalışan istemci kodu ax_reg işlevini bulur veitişlevini bulur ve yürürlükteki genel iş birimi içinde kalıcı çalışma yaptığını *dinamik olarak kaydettirmek* için bu işlevi çağırır. Bu ax_reg çağrısına yanıt olarak, veritabanının katıldığı WebSphere MQ kayıtlarında yer alan kayıtlar. Bu WebSphere MQ bağlantısındaki ilk ax_reg çağrısıysa, kuyruk yöneticisi aracısı xa_opençağrılarını çağırır.

Veritabanı istemci kodu, işleminizde çalışırken bu ax_reg çağrısını yapar; örneğin, bir SQL UPDATE çağrısı sırasında ya da veritabanının istemci API 'sında herhangi bir çağrı sorumlu olur.

Hata koşulları
XA dinamik kaydında, kuyruk yöneticisinde kafa karıştırıcı bir hata olasılığı vardır.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Ortak bir örnek, kuyruk yöneticinizi başlatmadan önce veritabanı ortam değişkenlerinizi düzgün olarak ayarlamayı unutmanız durumunda, kuyruk yöneticisinin `xa_open` ' e çağrılması başarısız olur. Genel iş birimi kullanılamaz.

Bunu önlemek için, kuyruk yöneticisini başlatmadan önce ilgili ortam değişkenlerini ayarladığınızdan emin olun. Veritabanı ürününüzün belgelerini ve “Db2yapılandırılıyor” sayfa 49, “Oracle' ın yapılandırılması” sayfa 51ve “Sybase yapılanışı” sayfa 56' da verilen öneriyi gözden geçirin.

Tüm veritabanı ürünleriyle, kuyruk yöneticisi kurtarma oturumunun bir parçası olarak (“Kişi XA kaynak yöneticisiyle kaybolduğunda dikkat edilecek noktalar” sayfa 59' ta açıklandığı gibi) kuyruk yöneticisi başlatıldığında `xa_open` bir kez çağrılıyor. Veritabanı ortam değişkenlerinizi yanlış ayarladığınızda, bu `xa_open` çağrısı başarısız olur, ancak kuyruk yöneticisinin başlatılamamasına neden olmaz. Bunun nedeni, veritabanı sunucusu tarafından veritabanı sunucusunun kullanılamadığını göstermek için veritabanı istemci kitaplığı tarafından aynı `xa_open` hata kodunun kullanılmasıdır. Kuyruk yöneticisinin bu veritabanını içeren genel iş birimleri dışındaki verileri işlemeye devam edebilmesi için, WebSphere MQ bu durumu ciddi bir hata olarak kabul etmez.

`xa_open` ' a sonraki çağrılar, WebSphere MQ bağlantısında (dinamik kayıt kullanılmıyorsa) ya da veritabanı istemci kodu tarafından WebSphere MQtarafından sağlanan `ax_reg` işleviyle (dinamik kayıt kullanılıyorsa) veritabanı istemci kodu çağrısı sırasında kuyruk yöneticisinden yapılan çağrılar.

Herhangi bir hata koşulunun **zamanlamasını** (ya da zaman zaman FFST raporları) dinamik kayıt kullanıp kullanmamanıza bağlıdır:

- Dinamik kayıt kullanıyorsanız, MQBEGIN çağrısının başarılı olması, ancak SQL UPDATE (ya da benzeri) veritabanı çağrısının başarısız olması gerekir.
- Dinamik kayıt kullanmayacaksanız, MQBEGIN çağrısını başarısız olur.

Uygulama ve kuyruk yöneticisi süreçlerinizde ortam değişkenlerinizin doğru olarak ayarlandığından emin olun.

XA çağrılarını özetleme
Bu, genel iş birimlerini denetleyen çeşitli MQI çağrılarının sonucu olarak, bir veritabanı istemci kitaplığındaki XA işlevlerine yapılan çağrıların bir listesini içerir. Bu, XA belirtiminde açıklanan protokolle ilgili tam bir açıklama değildir; kısa bir genel bakış olarak sağlanır.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Note that `xa_start` and `xa_end` calls are always called by WebSphere MQ code in the application process, whereas `xa_prepare`, `xa_commit`, and `xa_rollback` are always called from the queue manager agent process, amqzlaa0.

Bu çizelgede gösterilen `xa_open` ve `xa_close` çağrıları, uygulama sürecinden yapılır. Kuyruk yöneticisi aracısı işlemi, “Hata koşulları” sayfa 64'de açıklanan durumlarda `xa_open` ' ı çağırır.

Çizelge 7. XA işlev çağrılarının özeti

MQI çağrısı	**Devingen kayıt ile yapılan XA çağrıları**	**Devingen kayıt olmadan yapılan XA çağrıları**
İlk MQBEGIN	`xa_open`	`xa_open` `xa_start`
Sonraki MQBEGIN	XA çağrısı yok	`xa_start`

Çizelge 7. XA işlev çağrılarının özeti (devamı var)

MQI çağrısı	**Devingen kayıt ile yapılan XA çağrıları**	**Devingen kayıt olmadan yapılan XA çağrıları**

MQCMIT (yürürlükteki genel iş birimi sırasında çağrılmakta olan**bu olmadan** `ax_reg`)	XA çağrısı yok	`xa_end`
		`xa_prepare`
		`xa_commit`
		`xa_rollback`

MQCMIT (yürürlükteki genel iş birimi sırasında çağrılmakta olan**bununla** `ax_reg`)	`xa_end`
	`xa_prepare`
	`xa_commit`
	`xa_rollback`
		Geçerli değil. Dinamik olmayan kipte ax_reg 'e çağrılar yapılmayacak.

MQBACK (yürürlükteki genel iş birimi sırasında çağrılmakta olan**bu olmadan** `ax_reg`)	XA çağrısı yok	`xa_end` `xa_rollback`
MQBACK (yürürlükteki genel iş birimi sırasında çağrılmakta olan**bununla** `ax_reg`)	`xa_end` `xa_rollback`	Geçerli değil. Dinamik olmayan kipte ax_reg 'e çağrılar yapılmayacak.
MQDISC, burada MQCMT ya da MQBACK adı verildi. Bunlar yoksa, MQCMIT işlemi ilk olarak MQDISC sırasında yapılır.	`xa_close`	`xa_close`

Notlar:

1. MQCMIT için, `xa_prepare` başarılı olursa `xa_commit` çağrılır. Tersi durumda, `xa_rollback` çağrılır.

2. senaryo: Diğer yazılımlar koordinasyonu sağlar

2. senaryoda, bir dış hareket yöneticisi, genel iş birimlerini koordine eder, bunları hareket yöneticisinin API 'si denetimi altında başlatıp kesinleştirir. MQBEGIN, MQCMIT ve MQBACK fiilleri kullanılamıyor.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bu bölümde aşağıdakiler de dahil olmak üzere bu senaryo açıklanmaktadır:

- “Dış eşitleme noktası eşgüdümü” sayfa 66
- “ CICS' in kullanılması” sayfa 69
- “ Microsoft Transaction Server sunucusunun kullanılması (COM +)” sayfa 73

HP Integrity NonStop Server için IBM WebSphere MQ istemcisi, genel iş birimlerini koordine etmek için HP NonStop Transaction Management Facility (TMF) olanağını kullanabilir. Daha fazla bilgi için HP NonStop TMF ' nin kullanılmasıbaşlıklı konuya bakın.

Dış eşitleme noktası eşgüdümü

Genel iş birimi, dış X/Open XA uyumlu hareket yöneticisi tarafından da eşgüdümlü olarak kullanılabilir. Burada WebSphere MQ kuyruk yöneticisi katılır, ancak çalışma birimi koordinasyonu yoktur.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Dış hareket yöneticisi tarafından koordine edilen genel bir iş biriminde denetim akışı aşağıdaki gibidir:

1. Bir uygulama, dış eşitleme noktası eşgüdümcisine (örneğin, TXSeries) bir işlem başlatmak istediğini bildirir.
2. Sync Point Coordinator (Sync Point Coordinator), bilinen kaynak yöneticilerine (örneğin, WebSphere MQgibi) yürürlükteki işlemle ilgili bilgi verir.
3. Uygulama, geçerli işlemle ilişkili kaynak yöneticilerine çağrı gönderir. Örneğin, uygulama MQGET çağrılarını WebSphere MQolarak yayınlayabilirdi.
4. Uygulama, dış eşitleme noktası eşgüdümcisine bir kesinleştirme ya da geri alma isteği gönderir.
5. Tutarlılık noktası eşgüdümcüsü, her kaynak yöneticisine uygun çağrıları vererek, genellikle iki aşamalı kesinleştirme protokolleri kullanarak işlemi tamamlar.

WebSphere MQ katılımcılarının IBM WebSphere MQ ayrıntılı sistem gereksinimleriadresinde tanımlandığı işlemler için iki aşamalı kesinleştirme işlemi sağlayabilen, desteklenen dış eşitleme noktası koordinatörleri düzeyleridir.

Bu bölümün geri kalan kısmı, dış iş birimlerinin nasıl etkinleştirileceğini açıklar.

IBM WebSphere MQ XA anahtar yapısı
Dışarıdan eşgüdümlü bir iş birimine katılan her kaynak yöneticisi bir XA anahtar yapısı sağlamalıdır. Bu yapı, kaynak yöneticisinin yeteneklerini ve Sync Point koordinatörünün çağrılacak işlevleri tanımlar.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

IBM WebSphere MQ , bu yapının iki sürümünü sağlar:

- Durağan XA kaynak yönetimi için*MQRMIXASwitch*
- Dinamik XA kaynak yönetimi için*MQRMIXASwitchDynamic*

Statik ya da dinamik kaynak yönetimi arabirimini kullanıp kullanmayacağınızı belirlemek için işlem yöneticisi belgelerinize bakın. Bir hareket yöneticisinin desteklediği her yerde, devingen XA kaynak yönetimini kullanmanızı öneririz.

Bazı 64 bit işlem yöneticileri XA belirtimindeki *uzun* tipini 64 bit olarak ve bazıları ise 32 bit olarak kabul eder. WebSphere MQ her iki modeli de destekler:

- Hareket yöneticiniz 32 bit ise ya da işlem yöneticiniz 64 bit ise, ancak *long* tipini 32 bit olarak değerlendiriyorsa, Çizelge 8 sayfa 67içinde listelenen anahtar yükleme dosyasını kullanın.
- Hareket yöneticiniz 64 bit ise ve *long* tipini 64 bit olarak ele geçiriyorsa, Çizelge 9 sayfa 68içinde listelenen anahtar yükleme dosyasını kullanın.

uzun tipini 64 bit olarak kabul eden, bilinen 64 bit hareket yöneticilerinin listesi Çizelge 10 sayfa 68' te sağlanır. Hareket yöneticinizin kullandığı modelden emin değilseniz, işlem yöneticisi belgelerinize bakın.

Çizelge 8. XA anahtarı yükleme dosyası adları

Altyapı	**Anahtar yükleme dosyası adı (sunucu)**	**Anahtar yükleme dosyası adı (genişletilmiş işlemsel istemci)**
Pencereler	*mqmxa.dll*	*mqcxa.dll*
AIX (iş parçacıklı)	*libmqmxa.a*	*libmqcxa.a*
AIX (yivli)	*libmqmxa_r.a*	*libmqcxa_r.a*
HP-UX (yivsiz)	*libmqmxa.so*	*libmqcxa.so*
HP-UX (yivli)	*libmqmxa_r.so*	*libmqcxa_r.so*
Linux (iş parçacıklı)	*libmqmxa.so*	*libmqcxa.so*
Linux (yivli)	*libmqmxa_r.so*	*libmqcxa_r.so*

Çizelge 8. XA anahtarı yükleme dosyası adları (devamı var)		
Altyapı	**Anahtar yükleme dosyası adı (sunucu)**	**Anahtar yükleme dosyası adı (genişletilmiş işlemsel istemci)**
Solaris	*libmqmxa.so*	*libmqcxa.so*

Çizelge 9. Alternatif 64 bitlik XA anahtarı yükleme dosyası adları		
Altyapı	**Anahtar yükleme dosyası adı (sunucu)**	**Anahtar yükleme dosyası adı (genişletilmiş işlemsel istemci)**
AIX (iş parçacıklı)	*libmqmxa64.a*	*libmqcxa64.a*
AIX (yivli)	*libmqmxa64_r.a*	*libmqcxa64_r.a*
HP-UX (yivsiz)	*libmqmxa64.so*	*libmqcxa64.so*
HP-UX (yivli)	*libmqmxa64_r.so*	*libmqcxa64_r.so*
Linux (iş parçacıklı)	*libmqmxa64.so*	*libmqcxa64.so*
Linux (yivli)	*libmqmxa64_r.so*	*libmqcxa64_r.so*
Solaris	*libmqmxa64.so*	*libmqcxa64.so*

Çizelge 10. alternatif 64 bit anahtar yükleme dosyası gerektiren 64 bit işlem yöneticileri
İşlem Yöneticisi
Smokin

Bazı dış eşitleme noktası koordinatörleri (CICSdeğil), bir iş birimine katılan her kaynak yöneticisinin adını XA anahtar yapısının ad alanında yer alan bir iş birimine gereksinim duymasını gerektirir. WebSphere MQ kaynak yöneticisi adı, MQSeries_XA_RMI olarak adlanmlıdır.

Eşitleme noktası eşgüdümcüsü, WebSphere MQ XA anahtar yapısı bağlantılarının nasıl ilişkilendireceğini tanımlar. WebSphere MQ XA anahtar yapısının CICS ile bağlantılandırmayla ilgili bilgiler, “ CICS' in kullanılması” sayfa 69içinde sağlanır. WebSphere MQ XA anahtar yapısının diğer XA uyumlu eşitleme noktası koordinatörleriyle bağlantı kurmasıyla ilgili bilgi edinmek için, bu ürünlerle birlikte sağlanan belgelere bakın.

Tüm XA uyumlu eşitleme noktası koordinatörleriyle WebSphere MQ ' u kullanmak için aşağıdaki noktaları dikkate aldır:

- Sync Point eşgüdümcüsü tarafından herhangi bir xa_open çağrısında geçirilen xa_info yapısı, bir WebSphere MQ kuyruk yöneticisinin adını içerir. Ad, `MQCONN` çağrısına iletilen kuyruk yöneticisi adı ile aynı formu alır. xa_open çağrısına geçirilen ad boş bırakılırsa, varsayılan kuyruk yöneticisi kullanılır.

  Diğer bir seçenek olarak, xa_info yapısı, *TPM* ve *AXLIB* parametrelerinin değerlerini içerebilir. *TPM* parametresi, kullanılmakta olan hareket yöneticisini belirtir. Geçerli değerler şunlardır: CICS, SMOKIN ve ENCRINA. *AXLIB* parametresi, hareket yöneticisinin ax_reg ve ax_unreg işlevlerini içeren kitaplığın adını belirtir. Bu parametrelerle ilgili daha fazla bilgi için Genişletilmiş işlem istemcisinin yapılandırılmasıbaşlıklı konuya bakın. xa_info yapısı bu değiştirgelerden birini içeriyorsa, kuyruk yöneticisi adı varsayılan kuyruk yöneticisi kullanılmadıkça *QMNAME* değiştirgesinde belirtilir.
- Bir dış eşitleme noktası eşgüdümcisinin eşgüdümü eşgüdümü tarafından eşgüdümlü bir işleme yalnızca bir kuyruk yöneticisi katılabilir. Sync Point Coordinator (Sync Point Coordinator), kuyruk yöneticisine etkili bir şekilde bağlıdır ve bir kerede tek bir bağlantının desteklendiği kuralın konusuna tabidir.

- Dış eşitleme noktası eşgüdümcisine yapılan çağrıları içeren tüm uygulamalar, yalnızca dış eşgüdümcünün yönettiği işleme katılan kuyruk yöneticisine bağlanabilir (çünkü o kuyruk yöneticisine zaten etkin bir şekilde bağlandıkları için). Ancak, bu tür uygulamaların bağlantı tanıtıcısı almak için bir `MQCONN` çağrısı ve çıkmadan önce bir `MQDISC` çağrısı yayınlaması gerekir.
- Dış eşitleme noktası eşgüdümcüsü tarafından eşgüdümlü kaynak güncellemeleri olan bir kuyruk yöneticisi, dış eşitleme noktası eşgüdümcından önce başlamalıdır. Benzer şekilde, tutarlılık noktası eşgüdümcüsü kuyruk yöneticilikinden önce sona ermelidir.
- Dış eşitleme noktası eşgüdümcünüz olağandışı bir şekilde sona ererse, kuyruk yöneticinizi durdurun ve yeniden başlatın: ***önce*** , başarısızlığın zamanında kesinleştirilmemiş ileti işlemlerinin doğru biçimde çözülmesini sağlamak için Sync Point Coordinator olanağını yeniden başlatın.

CICS' in kullanılması

CICS , TXSeries' in öğelerinden biridir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

XA uyumlu (ve iki aşamalı kesinleştirme işlemi kullanan) TXSeries sürümleri şu adreste tanımlanır: IBM WebSphere MQ ayrıntılı sistem gereksinimleri

WebSphere MQ , diğer hareket yöneticilerini de destekler. Desteklenen yazılımların yürürlükteki listeleri için IBM WebSphere MQ ayrıntılı sistem gereksinimleri ' e bakın.

İki aşamalı kesinleştirme işleminin gereksinimleri

Requirements of the two-phase commit process when you use the CICS two-phase commit process with WebSphere MQ. Bu gereksinimler z/OSiçin geçerli değildir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Aşağıdaki gereksinimleri not edin:

- WebSphere MQ ve CICS aynı fiziksel makinede yer almalıdır.
- WebSphere MQ , bir WebSphere MQ MQI istemcisinden CICS ' i desteklemez.
- You must start the queue manager, with its name specified in the XAD resource definition stanza, ***önce*** you attempt to start CICS. Failure to do this will prevent you from starting CICS if you have added an XAD resource definition stanza for WebSphere MQ to the CICS region.
- Tek bir CICS bölgesinden bir kerede tek bir WebSphere MQ kuyruk yöneticisine erişilebilir.
- Bir CICS işlemi, WebSphere MQ kaynaklarına erişmeden önce bir `MQCONN` isteği yayınlamalıdır. `MQCONN` çağrısı, CICS bölgesi için XAD kaynak tanımlaması stanza 'nın XAOpen girişinde belirtilen WebSphere MQ kuyruk yöneticisinin adını belirtmelidir. Bu giriş boşsa, `MQCONN` isteğinin varsayılan kuyruk yöneticisini belirtmesi gerekir.
- WebSphere MQ kaynaklarına erişen bir CICS işlemi, CICS' e dönmeden önce hareketten bir `MQDISC` çağrısı yayınlamalıdır. Bunu yapmamanız, CICS uygulama sunucusunun bağlı olduğu, kuyrukların açık bırakıldığı anlamına gelebilir. Buna ek olarak, bir görev sonlandırma çıkışı kurmadıysanız (bkz. “Örnek görev sonlandırma çıkışı” sayfa 73), CICS uygulama sunucusu daha sonra olağandışı bir şekilde sona erebilir, belki de sonraki bir işlem sırasında.
- You must ensure that the CICS user ID (cics) is a member of the mqm group, so that the CICS code has the authority to call WebSphere MQ.

  Bir CICS ortamında çalışan işlemler için, kuyruk yöneticisi yetki yöntemlerini ve bağlamı belirleyerek aşağıdaki gibi saptanıyor:

  – The queue manager queries the user ID under which CICS runs the transaction. Bu, Nesne Yetkisi Yöneticisi tarafından denetlenen kullanıcı kimliğidir ve bağlam bilgileri için kullanılır.
  – İleti bağlamında, uygulama tipi MQAT_CICS ' tir.

– Bağlamdaki uygulama adı, CICS hareket adından kopyalanır.

Genel XA desteği
Genel XA, IBM iüzerinde desteklenmez. CICS ' i UNIX and Linux sistemlerinde WebSphere MQ ile bağlamanızı sağlamak için XA anahtar yükleme modülü sağlanır. Ayrıca, diğer hareket iletilerine ilişkin XA anahtarlarını geliştirmenizi sağlamak için örnek kaynak kod dosyaları da sağlanır.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Sağlanan anahtar yükleme modüllerinin adları şunlardır:

Çizelge 11. CICS uygulamaları için temel kod: XA kullanıma hazırlama yordamı

C (kaynak)	**C (exec)-aşağıdakilerden birini ekleyin:** **XAD.Stanza**
amqzscix.c	amqzsc- TXSeries for AIX, Sürüm 5.1, amqzsc- TXSeries for HP-UX, Sürüm 5.1 amqzsc- Sun Solaris, Sürüm 5.1için TXSeries
amqzscin.c	mqmc4swi - TXSeries for Windows, Sürüm 5.1

Building libraries for use with TXSeries for Multiplatforms
Use this information when building libraries for use with TXSeries for Multiplatforms.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Önceden oluşturulmuş anahtar yükleme dosyaları are shared libraries (called *DLL ' ler* on the Pencereler system) that you can use with CICS programs, which require a 2-phase commit transaction by using the XA protocol. Önceden oluşturulmuş bu kitaplıkların adlarıtableXA kullanıma hazırlama yordamındaEssential Essential code for CICS applicationsadlı çizelgede yer alıyor. Örnek kaynak kodu aşağıdaki dizinlerde de sağlanır:

Çizelge 12. Pencereler, UNIX and Linux işletim sistemlerinde kuruluş dizinleri

Altyapı	**Dizin**	**Kaynak Dosya**
UNIX and Linux	*MQ_INSTALLATION_PATH*/ samp/	amqzscix.c
Pencereler	*MQ_INSTALLATION_PATH*\Tools \c \ Örnekler	amqzscin.c

Burada*MQ_INSTALLATION_PATH* , IBM WebSphere MQprogramının kurulu olduğu dizindir.

Anahtar yükleme dosyasını örnek kaynağından oluşturmak için işletim sisteminiz için uygun olan yönergeleri izleyin:

AIX
Şu komutu verin:

```
export MQM_HOME=/usr/mqm
echo "amqzscix" > tmp.exp
xlc_r $MQM_HOME/samp/amqzscix.c -I/usr/lpp/cics/include -I$MQM_HOME/inc -e amqzscix -bE:tmp.exp
-bM:SRE -o amqzsc /usr/lpp/cics/lib/regxa_swxa.o -L$MQM_HOME/lib -L/usr/lpp/cics/lib -lcicsrt -lEncina
-lEncServer -lpthreads -lsarpc -lmqmcics_r -lmqmxa_r -lmqzi_r -lmqmcs_r
rm tmp.exp
```

Solaris

Şu komutu verin:

```
/opt/SUNWspro/bin/cc -s -l/opt/encina/include amqzscix.c -G -o amqzscix  -e
    CICS_XA_Init -LMQ_INSTALLATION_PATH/lib -L/opt/encina/lib
    -L/opt/dcelocal/lib /opt/cics/lib/reqxa_swxa.o
    -lmqmcics -lmqmxa -lmqzi -lmqmcs -lmqmzse -lcicsrt -lEncina -lEncSfs -ldce
```

HP-UX

Şu komutu verin:

```
cc -c -s -I/opt/encina/include MQ_INSTALLATION_PATH/samp/amqzscix.c -Aa +z -o  amqzscix.o ld -b
    -o amqzscix amqzscix.o /opt/cics/lib/regxa_swxa.o +e CICS_XA_Init \
    -LMQ_INSTALLATION_PATH/lib -L/opt/encina/lib -L/opt/cics/lib
    -lmqmxa_r -lmqzi_r -lmqmcs_r -lmqmzse -ldbm -lc -lm
```

Linux platformları

Şu komutu verin:

```
gcc -m32 -shared -fPIC -o amqzscix amqzscix.c
    \-IMQ_INSTALLATION_PATH/inc -I CICS_INSTALLATION_PATH/include
    \-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
    \-Wl,-rpath=/usr/lib -Wl,-rpath-link,/usr/lib -Wl,--no-undefined
    -Wl,--allow-shlib-undefined \-L CICS_LIB_PATH/regxa_swxa.o \-lpthread -ldl -lc
    -shared -lmqzi_r -lmqmxa_r -lmqmcics_r -ldl -lc
```

Pencereler

Aşağıdaki adımları izleyin:

1. En az aşağıdaki değişkenleri derleyerek amqzscin.obj oluşturmak için cl komutunu kullanın:

   ```
   cl.exe -c -IEncinaPath\include -IMQ_INSTALLATION_PATH\include -Gz -LD amqzscin.c
   ```

2. Aşağıdaki satırları içeren `mqmc1415.def`adlı bir modül tanımlama dosyası yaratın:

   ```
   LIBRARY MQMC4SWI
   EXPORTS
   CICS_XA_Init
   ```

3. En az aşağıdaki seçeneği kullanarak bir dışa aktarma dosyası ve içe aktarma kitaplığı oluşturmak için **lib** komutunu kullanın:

   ```
   lib -def:mqmc4swi.def -out:mqmc4swi.lib
   ```

   lib komutu başarılı olursa, bir `mqmc4swi.exp` dosyası da oluşturulur.

4. En az aşağıdaki seçeneği kullanarak `mqmc4swi.dll` oluşturmak için bağlantı komutunu kullanın:

   ```
   link.exe -dll -nod -out:mqmc4swi.dll
       amqzscin.obj CicsPath\lib\regxa_swxa.obj
       mqmc4swi.exp mqmcics4.lib
      CicsPath\lib\libcicsrt.lib
      DcePath\lib\libdce.lib DcePath\lib\pthreads.lib
      EncinaPath\lib\libEncina.lib
      EncinaPath\lib\libEncServer.lib
      msvcrt.lib kernel32.lib
   ```

IBM WebSphere MQ XA desteği ve Tuxedo

PencerelerüzerindeIBM WebSphere MQ ,UNIX and Linux sistemleri, xa_start içinde TUXEDO eşgüdümlü XA uygulamalarını süresiz olarak engelleyebilir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bu durum yalnızca, tek bir genel hareket girişindeki Tuxedo tarafından eşgüdümlü iki ya da daha çok işlem, aynı hareket dalı tanıtıcısı (XID) kullanılarak IBM WebSphere MQ ' a erişme girişiminde bulunulduğunda ortaya çıkabilir. Smokin, genel harekette her bir işlemi IBM WebSphere MQile kullanılacak farklı bir XID verirse, bu gerçekleşemez.

Sorunu önlemek için, kendi Tuxedo sunucusu grubu içinde, tek bir genel hareket tanıtıcısı (gtrid) altında IBM WebSphere MQ ' e erişen Tuxedo içindeki her bir uygulamayı yapılandırın. Processes in the same server group use the same XID when accessing resource managers on behalf of a single gtrid, and are therefore vulnerable to blocking in xa_start in IBM WebSphere MQ. Farklı sunucu gruplarındaki işlemler, kaynak yöneticilerine erişirken ayrı XID ' ler kullanır; bu nedenle, hareket işlerini IBM WebSphere MQiçinde diziselleştirmek zorunda kalmayın.

CICS iki aşamalı kesinleştirme işleminin etkinleştirilmesi
CICS ' in MQI çağrılarını içeren işlemleri koordine etmek üzere iki aşamalı bir kesinleştirme işlemi kullanmasını sağlamak için CICS bölgesine bir CICS XAD kaynak tanımlaması stanza girişi ekleyin. Bu konu, z/OSiçin geçerli değildir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Here is an example of adding an XAD stanza entry for WebSphere MQ for Pencereler, where <Drive> is the drive where WebSphere MQ is installed (for example, D:).

```
cicsadd -cxad -r<cics_region> \
     ResourceDescription="MQM XA Product Description" \
     SwitchLoadFile="<Drive>:\Program Files\IBM\WebSphere MQ\bin\mqmc4swi.dll" \
     XAOpen=<queue_manager_name>
```

Genişletilmiş işlemsel istemciler için, mqcc4swi.dllanahtar yükleme dosyasını kullanın.

Here is an example of adding an XAD stanza entry for WebSphere MQ for UNIX and Linux systems, where *MQ_INSTALLATION_PATH* represents the high-level directory in which WebSphere MQ is installed:

```
cicsadd -cxad -r<cics_region> \
      ResourceDescription="MQM XA Product Description" \
      SwitchLoadFile="MQ_INSTALLATION_PATH/lib/amqzsc" \
      XAOpen=<queue_manager_name>
```

Genişletilmiş işlemsel istemciler için, amqczsc anahtar yükleme dosyasını kullanın.

cicsadd komutunun kullanılmasıyla ilgili bilgi edinmek için, altyapınıza ilişkin *CICS Administration Reference*ya da *CICS Administration Guide* belgesine bakın.

WebSphere MQ çağrıları, bir CICS işlemine dahil edilebilir ve WebSphere MQ kaynakları, CICStarafından yönlendirilmiş olarak kesinleştirilir ya da geri alınır. Bu destek, istemci uygulamaları için kullanılamaz.

You ***gerekir*** issue an MQCONN from your CICS transaction in order to access WebSphere MQ resources, followed by a corresponding MQDISC on exit.

CICS kullanıcı çıkışlarının etkinleştirilmesi
A CICS user exit *nokta* (normally referred to as a *kullanıcı çıkışı*) is a place in a CICS module at which CICS can transfer control to a program that you have written (a user exit *program*), and at which CICS can resume control when your exit program has finished its work.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bir CICS kullanıcı çıkışı kullanmadan önce, altyapınıza ilişkin *CICS Administration Guide* adlı kılavuzunu okuyun.

Örnek görev sonlandırma çıkışı
WebSphere MQ , bir CICS görevi sonlandırma çıkışı için örnek kaynak kodu sağlar.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Örnek kaynak kodu aşağıdaki dizinlerde yer alıyor:

Çizelge 13. CICS görev sonlandırma çıkışları

Altyapı	**Dizin**	**Kaynak Dosya**
UNIX and Linux sistemleri	*MQ_INSTALLATION_PATH*/samp	amqzscgx.c
Pencereler	*MQ_INSTALLATION_PATH*\Tools \c \ Örnekler	amqzscgn.c

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Örnek görev sonlandırma çıkışa ilişkin oluşturma yönergeleri, her kaynak dosyasının üst kısmına yakın yorumlarda yer alır.

Bu çıkış, olağan dışı görev sonlandırmasında (herhangi bir eşitleme noktası alındıktan sonra) CICS tarafından çağrılır. Çıkış programında kurtarılabilir bir işe izin verilmez.

Bu işlevler yalnızca, CICS sürümünün XA arabirimini desteklediği WebSphere MQ ve CICS bağlamında kullanılır. CICS , bu kitaplıkları "programlar" ya da "kullanıcı çıkışları"olarak belirtir.

CICS has a number of user exits and amqzscgx, if used, is defined and enabled on CICS as the "Görev sonlandırma kullanıcı çıkışı (UE014015)", that is, exit number 15.

Görev sonlandırma çıkışı CICStarafından çağrıldığında, CICS görevin sonlandırma durumuna ilişkin WebSphere MQ ' i önceden bilgilendirdi ve WebSphere MQ uygun işlemi (kesinleştirme ya da geriye işleme) aldı. Çıkışta, temizlemek için bir MQDISC komutu vermeniz gerekir.

Bir görev sonlandırma çıkışını kullanmak üzere CICS sisteminizi kurmak ve yapılandırmak için tek bir amaç, sisteminizi hatalı uygulama kodunun bazı sonuçlarına karşı korumaktan başka bir amaç değildir. Örneğin, CICS işleminiz önce MQDISC çağrılmadan olağan dışı sona ererse ve herhangi bir görev sonlandırma çıkışı kurulu değilse, CICS bölgesinin kurtarılamayan bir hatasını görebilirsiniz (10 saniye içinde). Bunun nedeni, cicsas işleminde çalışan WebSphere MQ' nun sağlık iş parçacığından, temizlenmek ve geri dönmek için zaman verilmeyeceğinden kaynaklanır. Belirtiler, cicsas işleminin hemen sona ermesi, FFT raporlarını /var/mqm/errors ' e ya da Pencerelerüzerindeki eşdeğer konuma yazması olabilir.

Microsoft Transaction Server sunucusunun kullanılması (COM +)

COM + (Microsoft Transaction Server), kullanıcıların iş mantığı uygulamalarını tipik bir orta katman sunucusunda çalıştırmasına yardımcı olmak için tasarlanmıştır.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Önemli bilgi için Yalnızca Windows 'ta birincil kuruluşla kullanılabilen özellikler konusuna bakın.

COM+ divides work up into *etkinlikler*, which are typically short independent chunks of business logic, such as *hesap A 'dan hesap B' ye aktarma fonları*. COM + nesne yönüne ve özellikle COM ' a yoğun bir şekilde dayanır; gevşek bir COM + etkinliği, COM (iş) nesnesi tarafından temsil edilir.

COM +, işletim sisteminin tümleşik bir parçasıdır. To use COM+ on Pencereler 2000 and Pencereler XP, you need Hotfix Q313582 (also known as COM+ Rollup Package 19.1).

COM +, iş nesnesi denetimcisine üç hizmet sağlar ve iş nesnesi programcısından gelen endişelerin çoğunu ortadan kaldırır:

- Hareket yönetimi
- Güvenlik
- Kaynak havuzu oluşturma

COM + ile genellikle COM + ön uç koduyla, COM + içinde tutulan nesneler ve veritabanı gibi arka uç hizmetleri (WebSphere MQ) ile COM + iş nesnesi ve arka uç arasında bir köprü kullanıyorsunuz.

Ön uç kodu, bağımsız bir program olabilir ya da Microsoft Internet Information Server (IIS) tarafından barındırılan bir Etkin Sunucu Sayfası (ASP) olabilir. Ön uç kodu, COM + ve iş nesneleriyle aynı bilgisayarda, COM ile bağlantı kurabilmektedir. Diğer bir seçenek olarak, ön uç kodu, DCOM aracılığıyla bağlantıyla farklı bir bilgisayarda olabilir. Farklı durumlarda aynı COM + iş nesnesine erişmek için farklı istemciler kullanabilirsiniz.

Arka uç kodu, COM + ve iş nesneleri ile aynı bilgisayarda ya da WebSphere MQ destekli iletişim kurallarından herhangi biri aracılığıyla bağlantı içeren farklı bir bilgisayarda olabilir.

Genel iş birimleri sona eriyor

Kuyruk yöneticisi, önceden yapılandırılmış bir etkinlik dışı durum aralığından sonra genel iş birimlerinin süresini bitirecek şekilde yapılandırılabilir.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

Bu davranışı etkinleştirmek için aşağıdaki ortam değişkenlerini ayarlayın:

- *AMQ_TRANSACTION_EXPIRY_RESCAN*= < milisaniye cinsinden yeniden tarama aralığı >
- *AMQ_XA_TRANSACTION_IFADESI*= < milisaniye cinsinden zamanaşımı aralığı >

Uyarı: Ortam değişkenleri, XA Belirtiminin 6-4. Tablolarında yalnızca *Boşta Duran* durumunda olan hareketleri etkiler. Bu, herhangi bir uygulama iş parçacığıyle ilişkili olmayan, ancak dış İşlem Yöneticisi yazılımının henüz **`xa_prepare`** işlev çağrısını çağırmadığı işlemlerdir.

Dış hareket yöneticileri yalnızca, hazırlanmış, kesinleştirilen ya da geriye işlenen işlemlerden oluşan bir günlük alımlarıyla devam eder. Dış hareket yöneticisi herhangi bir nedenle aşağıya inerse, geri dönüş işlemi için hazırlanmış, kesinleştirilmiş ve geriye işlenen işlemleri tamamlar; ancak henüz hazırlıklanmış olan tüm etkin işlemler artık yalnız kaldı. Bunu önlemek için, *AMQ_XA_TRANSATION_EXPONY* ' u bir uygulama ile MQI işlemsel API çağrıları yapan bir uygulama arasında beklenen aralık için izin verecek şekilde ayarlayın ve diğer kaynak yöneticileri üzerinde işlemsel çalışma gerçekleştirmeyi gerçekleştirin.

To ensure a timely cleanup after the *AMQ_XA_TRANSATION_EXPONY* expires, set the *AMQ_TRANSACTION_EXPIRY_RESCAN* value to a lower value than the *AMQ_XA_TRANSATION_EXPONY* interval, ideally so that the rescan occurs more than once within the *AMQ_XA_TRANSATION_EXPONY* interval.

Kurtarma işlemi birimi

WebSphere MQ for z/OS kurtarma işlemleri birimi sağlar. Bu özellik, aynı kuyruk paylaşım grubundaki (QSG) başka bir kuyruk yöneticisine bağlanıldığında, 2 aşamalı kesinleştirme hareketlerinin ikinci evresinin (örneğin, kurtarma sırasında) çalıştırılıp sürmeyeceğini yapılandırmanızı sağlar.

Not: Bu konu, IBM MQ Version 8.0 ve sonraki sürümlerde de kullanılabilir. Ancak, "Sürümü değiştir" liste kutusunu kullanarak daha sonraki bir sürüme geçemezsiniz. Daha sonraki bir sürümle konuya gitmek için, tarayıcınızdaki URL kutusunda sürüm numarasını düzenleyin.

WebSphere MQ for z/OS V7.0.1 ve sonraki düzeyler, kurtarma atma birimini destekler.

Kurtarma işlemi birimi
Kurtarma yok etme birimi, bir uygulamanın bağlantısıyla ve ardından başlatıldığı herhangi bir işlemle ilgilidir. İki olası kurtarma işlemi birimi vardır.

- Bir grup kurtarma işlemi, bir işlemsel uygulamanın mantıksal olarak kuyruk paylaşım grubuna bağlı olduğunu ve belirli bir kuyruk yöneticisine benzeşimi olmadığını belirtir. Kesinleştirme işleminin phase-1 işlemini tamamlamış olan 2 aşamalı kesinleştirme hareketlerinin, QSG içindeki herhangi bir kuyruk yöneticisine bağlandıkları zaman sorgulanabilir ve çözülmüş olabilen, sorgulanabilen 2 aşamalı kesinleştirme işlemleridir. Kurtarma senaryolarında bu, hareket eşgüdümcünün aynı kuyruk yöneticisine yeniden bağlanmak zorunda olmadığı anlamına gelir; bu da kullanılamayabilir.
- Bir QMGR kurtarma düzeni birimi, bir uygulamanın, bağlı olduğu kuyruk yöneticisine doğrudan benzeşimi olduğunu ve bu uygulamanın başladığı tüm hareketlerin de bu yok etme durumunu belirtir.

  Bir kurtarma senaryosında, hareket koordinatörünün aynı kuyruk yöneticisine yeniden bağlanması ve kuyruk yöneticisinin bir kuyruk paylaşım grubuna ait olup olmadığına bakılmaksızın, belirsiz hareketleri sorgulamak ve çözmesi gerekir.

Kullanılacak programlama diline karar verme

Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

IBM WebSphere MQ , aşağıdaki programlama yordamlarıyla ilgili destek sağlar:

- C
- Visual Basic (yalnızca Windows sistemleri)
- COBOL

Bu diller, ileti kuyruklama hizmetlerine erişmek için ileti kuyruğu arabirimini (MQI) kullanır. Bu dillere ilişkin destek hakkında daha fazla bilgi için bkz. “Using procedural languages with WebSphere MQ” sayfa 75.

IBM WebSphere MQ aşağıdakiler için destek sağlar:

- .NET
- ActiveX
- C++
- Java
- JMS

These languages use the IBM WebSphere MQ Object Model, which provides classes that provide the same functionality as WebSphere MQ calls and structures, but that are a more natural way of programming in an object-oriented environment. IBM WebSphere MQ Nesne Modeli 'ni kullanan bazı diller, ileti kuyruğu arabiriminde (MQI) kullanılmıyorsa ek işlevler sağlar. Bu dillere ilişkin destek hakkında daha fazla bilgi için bkz. “WebSphere MQile nesne odaklı programlama” sayfa 76.

Using procedural languages with WebSphere MQ

Uygulamalarınızı seçtiğiniz dilde nasıl yazılabilmeye ilişkin ayrıntılı bilgi için aşağıdaki bağlantıları kullanın:

- “C içinde kodlama” sayfa 79
- “Visual Basic Kodlamasi” sayfa 83
- “COBOL içinde kodlama” sayfa 82

Yordamsal dillere ilişkin çağrı arabirimine genel bakış için Çağrı açıklamalarıbaşlıklı konuya bakın. Bu konu, MQI çağrılarının bir listesini içerir ve her bir çağrı size bu dillerin her birindeki çağrıların kodlarının nasıl kodlacağını gösterir.

WebSphere MQ , uygulamalarınızı yazmanıza yardımcı olacak veri tanımlama dosyaları sağlar. Tam açıklama için bkz. “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77.

Programlarınızın hangi dili kodlayacağını seçebilirseniz, programlarınızın işleyeceği iletilerin uzunluk üst sınırını göz önünde bulundurun. Programlarınız yalnızca bilinen bir uzunluk üst sınırını işleyecekse, bunları

desteklenen programlama dillerinin herhangi birinde kodlayabilirsiniz. Ancak, programların işlemek için sahip olacağı iletilerin uzunluk üst sınırını bilmiyorsanız, seçtiğiniz dil bir CICS, IMSya da toplu iş uygulaması mı yazıyorsanız bağlı olacaktır:

IMS ve toplu iş
Programları C, PL/I ya da çevirici dilinde kodlamak için, bu dillerin isteğe bağlı bellek miktarlarını elde etmek ve serbest bırakmak için sunduğu olanakların kullanılmasını sağlar. Diğer bir seçenek olarak, programlarınızı COBOL ' de kodlayabilrsiniz, ancak depolama alanını almak ve serbest bırakmak için çevirici dili, PL/I ya da C alt kenar çizgileri kullanabilirsiniz.

CICS
Programları, CICStarafından desteklenen herhangi bir dilde kodlayın. EXEC CICS arabirimi, gerekirse, belleği yönetmeye ilişkin çağrıları sağlar.

WebSphere MQile nesne odaklı programlama

IBM WebSphere MQ Nesne Modeli 'ni kullanan bazı diller ve programlama çerçeveleri, ileti kuyruğu arabiriminde (MQI) kullanılabilecek ek işlevler sağlar. IBM WebSphere MQ Nesne Modeli tarafından sağlanan sınıflar, yöntemler ve özelliklerle ilgili ayrıntılar için bkz. “IBM WebSphere MQ Nesne Modeli” sayfa 84.

.NET
See .NET ' in kullanılması for information about coding .NET programs using the WebSphere MQ .NET classes. Message Service Clients for C/C++ and .NET , Java Message Service (JMS) ile aynı arabirim kümesine sahip XMS adında bir uygulama programlama arabirimi (API) sağlar. API.

C++
IBM WebSphere MQ , WebSphere MQ nesnelerine eşdeğeri olan C++ sınıflarını ve dizi veri tiplerine eşdeğer bazı ek sınıfları sağlar. Bu, MQI aracılığıyla olmayan bazı özellikleri sağlar. C + + içinde WebSphere MQ Nesne Modeli 'ni kullanan kodlama programlarına ilişkin bilgi için C++ kullanılması ' e bakın. Message Service Clients for C/C++ and .NET , Java Message Service (JMS) ile aynı arabirim kümesine sahip XMS adlı bir uygulama programlama arabirimi (API) sağlar. API.

Java
Java 'da WebSphere MQ Nesne Modeli 'ni kullanarak kodlama programlarına ilişkin bilgi için Java 'nın kullanılması ' e bakın. For information about the differences between IBM WebSphere MQ classes for Java and IBM WebSphere MQ classes to help you decide which to use, see “JMS için Java ya da IBM WebSphere MQ sınıfları için IBM WebSphere MQ sınıfları kullanmalı mıyım?” sayfa 85.

JMS
Websphere MQ , Java Message Service (JMS) belirtimini gerçekleştiren sınıfları da sağlar. JMS için Websphere MQ sınıflarına ilişkin ayrıntılar için bkz. Java 'nın kullanılması. Hangi kullanacağınız konusunda karar vermenize yardımcı olacak Java ve IBM WebSphere MQ sınıflarına ilişkin IBM WebSphere MQ sınıfları arasındaki farklara ilişkin bilgi edinmek için bkz. “JMS için Java ya da IBM WebSphere MQ sınıfları için IBM WebSphere MQ sınıfları kullanmalı mıyım?” sayfa 85.

Message Service Clients for C/C++ and .NET , Java Message Service (JMS) ile aynı arabirim kümesine sahip XMS adında bir uygulama programlama arabirimi (API) sağlar. API.

ActiveX
WebSphere MQ ActiveX yaygın olarak MQAX olarak bilinir. The MQAX is included as part of WebSphere MQ for Pencereler.Support for ActiveX has been stabilized at the WebSphere MQ Version 6.0 level. To exploit features introduced to WebSphere MQ later than Version 6.0, consider using .NET instead. Refer to Bileşen Nesne Modeli Arabiriminin Kullanılması (WebSphere MQ Automation Classes for ActiveX) for information about coding programs using the WebSphere MQ Object Model in ActiveX.

İlgili kavramlar
Teknik genel bakış
“Uygulamaların geliştirilmesi” sayfa 7
IBM WebSphere MQ , iş süreçlerinizi desteklemek için gereksinim duyardığınız iletileri göndermek ve almak için uygulamalar geliştirebileceğiniz çeşitli yollar sağlar. Kuyruk yöneticilerinizi ve ilgili kaynaklarınızı yönetmek için de uygulamalar geliştirebilirsiniz.

“Uygulama geliştirme kavramları” sayfa 7

IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

İlgili başvurular

Uygulama geliştirme başvurusu

IBM WebSphere MQ veri tanımlama dosyaları

IBM WebSphere MQ , uygulamalarınızı yazmanıza yardımcı olacak veri tanımlama dosyaları sağlar.

Veri tanımlama dosyaları aşağıdaki gibi de bilinir:

Dil	Veri tanımlamaları
C	Dosyaları ya da üstbilgi dosyalarını dahil et
Visual Basic	Modül dosyaları (yalnızca 32 bitlik sürümler)
COBOL	Dosyaları kopyala
Çevirici	Makrolar
PL/I	İçerme dosyaları

Kanal çıkışlarını yazmanıza yardımcı olacak veri tanımlama dosyaları, WebSphere MQ COPY, HEADER, include ve module filesiçinde açıklanır.

Kurulabilir hizmet çıkışlarını yazmanıza yardımcı olacak veri tanımlama dosyaları “Kullanıcı çıkışları, API çıkışlar ve WebSphere MQ kurulabilir hizmetleri” sayfa 357içinde açıklanmıştır.

C + + üzerinde desteklenen veri tanımlama dosyaları için bkz. C++ kullanılması.

Veri tanımlama dosyalarının adları CMQ önekine ve programlama diline göre belirlenen bir sonektir:

Sonek	Dil
a	Çevirici dili
b	Visual Basic
c	C
l	COBOL (kullanıma hazırlanmamış değerler)
p	PL/I
v	COBOL (varsayılan değerler kümesi ile)

Kuruluş kitaplığı

The name **thlqual** is the high-level qualifier of the installation library on z/OS.

Bu konuda, aşağıdaki başlıklar altında WebSphere MQ veri tanımlama dosyaları tanıtılır:

- “C dili, dosyaları içerir” sayfa 77
- “Visual Basic modül dosyaları” sayfa 78
- “COBOL kopya dosyaları” sayfa 78

C dili, dosyaları içerir

The WebSphere MQ C include files are listed in C üstbilgi dosyaları. Bunlar, aşağıdaki dizinlere ya da kitaplıklara kurulur:

Altyapı	Kuruluş dizini ya da kitaplığı
UNIX altyapıları	*MQ_INSTALLATION_PATH*/inc/
Windows sistemleri	*MQ_INSTALLATION_PATH*\Tools\c\include

Burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Not: UNIX altyapılarında, içerme dosyaları sembolik olarak `/usr/include`deile bağlantılıdır.

Dizinlerin yapısı hakkında daha fazla bilgi için Planning file system support başlıklı konuya bakın.

Visual Basic modül dosyaları

WebSphere MQ for Windows , dört Visual Basic modülü dosyası sağlar.

Bunlar Visual Basic modül dosyaları içinde listelenir ve içinde kurulur.

```
MQ_INSTALLATION_PATH\Tools\Samples\VB\Include
```

COBOL kopya dosyaları

COBOL için, WebSphere MQ , adlandırılmış değişmezleri içeren ayrı kopya dosyaları ve her bir yapı için iki kopya dosyası sağlar.

Her bir yapı için iki kopya dosyası vardır. Bunun nedeni, her ikisinin de başlangıç değerleri olması ve başlangıç değerleri olmaksızın sağlanması:

- Bir COBOL programının ÇALIŞMA-STORAGE BöLüMüNDE, yapı alanlarını varsayılan değerlere ilk kullanıma hazırlatan dosyaları kullanın. Bu yapılar, V harfi (değerler) ile suffixed adlarına sahip kopya dosyalarda tanımlanır.
- Bir COBOL programının LINKUB BöLMESINDE, başlangıç değerleri olmayan yapıları kullanın. Bu yapılar, L (bağ) harfi ile suffixed adlarına sahip olan kopyalarda tanımlanır.

WebSphere MQ COBOL kopya dosyaları, COBOL COPY dosyalarıiçinde listelenir. Bunlar aşağıdaki dizinlere kurulur:

Altyapı	Kuruluş dizini ya da kitaplığı
Diğer UNIX altyapıları	*MQ_INSTALLATION_PATH*/inc/
Pencereler	*MQ_INSTALLATION_PATH*\Tools\cobol\copybook (Micro Focus COBOL için) *MQ_INSTALLATION_PATH*\Tools\cobol\copybook\VAcobol (IBM VisualAge COBOL için)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Programınıza yalnızca gereksinim duyardığınız dosyaları ekleyin. Bunu bir level-01 bildiriminden sonra bir ya da daha çok COPY deyimiyle gerçekleştirin. Bu, gerekirse bir programdaki yapıların birden çok sürümünü de içerebileceğiniz anlamına gelir. CMQV ' nin büyük bir dosya olduğunu unutmayın.

CMQMDV kopyalama dosyasını içermek için COBOL kodunun bir örneği aşağıda yer alır:

```
01 MQM-MESSAGE-DESCRIPTOR.
   COPY CMQMDV.
```

Her yapı bildirimi bir level-01 öğesiyle başlar; level-01 bildirimini kodlayarak, yapı bildiriminin geri kalanına kopyalanacak bir COPY deyiminin ardından, yapının birkaç örneğini bildirebilirsiniz. Uygun örneğe başvurmak için IN anahtar sözcüğünü kullanın.

Aşağıda, CMQMDV ' nin iki eşgörünümünü içermek için COBOL kodu örneği bulunmaktadır:

```
* Declare two instances of MQMD
 01  MY-CMQMD.
     COPY CMQMDV.
 01  MY-OTHER-CMQMD.
     COPY CMQMDV.
*
* Set MSGTYPE field in MY-OTHER-CMQMD
     MOVE MQMT-REQUEST TO MQMD-MSGTYPE IN MY-OTHER-CMQMD.
```

Yapıları 4 baytlık sınırlarla hizalayın. If you use the COPY statement to include a structure following an item that is not the level-01 item, ensure that the structure is a multiple of 4-bytes from the start of the level-01 item. Bunu yapmazsanız, uygulamanızın performansını düşürebilirsiniz.

Yapılar, MQI ' da kullanılan veri tipleriiçinde açıklanmıştır. Yapılardaki alanların tanımları, öneki olmayan alanların adlarını gösterir. COBOL programlarında, alan adlarına, COBOL bildirimlerinde gösterildiği gibi, yapının adını bir tire işareti ile önek olarak ekleyin. Yapı kopyalama dosyalarındaki alanlar bu şekilde önekli olur.

Yapı kopyalama dosyalarındaki bildirimlerdeki alan adları büyük harfle karakterdir. Bunun yerine küçük harf ya da küçük harf ya da küçük harf kullanabilirsiniz. Örneğin, MQGMO yapısının *StrucId* alanı COBOL bildiriminde ve kopyalama dosyasında MQGMO-STRUCID olarak gösterilir.

V-sonek yapıları, tüm alanlar için başlangıç değerleriyle bildirilir; bu nedenle, yalnızca gereken değerin başlangıç değerinden farklı olduğu alanları ayarlamanız gerekir.

C içinde kodlama

C içindeki WebSphere MQ programları kodlanırken aşağıdaki bölümlerdeki bilgileri not edin.

MQI çağrılarına ilişkin değiştirgeler

yalnızca giriş ve MQHCONN, MQHOBJ, MQHMSG ya da MQHOMEN tipinde olan değiştirgeler değer tarafından geçirilir; diğer tüm parametreler için, değiştirgenin *adresi* değeri değer olarak geçirilir.

Bir işlev çağrıldığında, adresle geçirilen tüm parametrelerin belirtilmesi gerekir. Belirli bir parametrenin gerekli olmadığı durumlarda, parametre verilerinin adresi yerine, işlev çağırımında parametre olarak boş değerli bir gösterge belirtilebilir. Bunun olası olduğu parametreler, çağrı açıklamalarında tanımlanır.

İşleve değer olarak bir parametre döndürülmez; C terminolojisinde, bu, tüm işlevlerin geçersiz döndürdüğü anlamına gelir.

İşleve ilişkin öznitelikler MQENTRY makro değişkeniyle tanımlanır; bu makro değişkeninin değeri ortama bağlıdır.

Tanımlanmamış veri tipine sahip parametreler

MQGET, MQPUT ve MQPUT1 işlevlerinin her biri, tanımlı olmayan bir veri tipine sahip bir *Buffer* parametresine sahip. Bu parametre, uygulamanın ileti verilerini göndermek ve almak için kullanılır.

Bu sıralamayı içeren değiştirgeler, C örneklerinde MQBYTE dizisi olarak gösterilir. Parametreleri bu şekilde bildirebilirsiniz, ancak genellikle iletilerde verilerin yerleşim düzenini tanımlayan yapı olarak bildirilmesi

daha kolay olur. İşlev parametresi bir işaretçi-void değeri olarak bildirilir ve bu nedenle, herhangi bir verinin adresi, işlev çağırımı üzerinde parametre olarak belirtilebilir.

Veri tipleri

Tüm veri tipleri `typedef` deyimiyle tanımlanır.

Her veri tipi için, ilgili gösterge veri tipi de tanımlıdır. İşaretçi veri tipinin adı, bir işaretçiyi göstermek için P harfiyle önekli olan temel ya da yapı veri tipinin adıdır. İşaretçinin öznitelikleri MQPOINTER makro değişkeni tarafından tanımlanır; bu makro değişkeninin değeri ortama bağlıdır. Aşağıdaki kod işaretçi veri tiplerinin nasıl bildirileceğini göstermektedir:

```
#define MQPOINTER                   /* depends on environment */
...
typedef MQLONG  MQPOINTER PMQLONG;  /* pointer to MQLONG */
typedef MQMD   MQPOINTER PMQMD;     /* pointer to MQMD */
```

İkili dizgileri işleme

İkili veri dizgileri, MQBYTEn veri tiplerinden biri olarak bildirilir.

Whenever you copy, compare, or set fields of this type, use the C functions `memcpy`, `memcmp`, or `memset`:

```
#include <string.h>
#include "cmqc.h"

MQMD MyMsgDesc;

memcpy(MyMsgDesc.MsgId,           /* set "MsgId" field to nulls    */
       MQMI_NONE,                 /* ...using named constant      */
       sizeof(MyMsgDesc.MsgId));

memset(MyMsgDesc.CorrelId,        /* set "CorrelId" field to nulls */
       0x00,                      /* ...using a different method   */
       sizeof(MQBYTE24));
```

Bu, MQBYTE24olarak bildirilen verilerle doğru bir şekilde çalışmadığından, strcpy, strcmp, strncpy ya da strncmp dizgisiyle ilgili dizilimi kullanmayın.

Karakter dizgillerini işleme

Kuyruk yöneticisi uygulamaya karakter verisi döndürdüğünde, kuyruk yöneticisi karakter verilerini her zaman, alanın tanımlı uzunluğuna sahip olacak şekilde boşluklarla doldurur. Kuyruk yöneticisi boş karakterle biten dizgileri döndürmüyor, ancak bunları girişinizde kullanabilirsiniz. Bu nedenle, bu tür dizgileri kopyalarken, karşılaştırırken ya da bitiştirirken, strncpy, strncmp ya da strncat dizgi işlevlerini kullanın.

Dizginin boş değerli (strcpy, strcmp ve strcat) sona erdirilmesini gerektiren dizgi işlevlerini kullanmayın. Ayrıca, dizilimin uzunluğunu belirlemek için strlen işlevini kullanmayın; alanın uzunluğunu belirlemek için büyüklük işlevi yerine kullanılır.

Yapılara ilişkin ilk değerler

İçerme dosyası <cmqc.h>, bu yapıların somut örneklerini bildirirken yapılara ilişkin başlangıç değerlerini sağlamak için kullanabileceğiniz çeşitli makro değişkenlerini tanımlar. Bu makro değişkenlerinin adları MQxxx_default biçiminin adlarına sahiptir; burada MQxxx, yapının adını gösterir. Bunları aşağıdaki gibi kullanın:

```
MQMD   MyMsgDesc = {MQMD_DEFAULT};
MQPMO  MyPutOpts = {MQPMO_DEFAULT};
```

Bazı karakter alanları için, geçerli olan MQI (örneğin, *StrucId* alanları için ya da MQMD ' deki *Format* alanı için) belirli değerleri tanımlar. Geçerli değerlerin her biri için iki makro değişkeni sağlanır:

- Bir makro değişkeni, değeri, alanın tanımlı uzunluğuna tam olarak uyan örtük boş (null) boş değer dışında bir uzunluğa sahip bir dizgi olarak tanımlar. Örneğin, simge boş bir karakteri temsil eder:

  ```
  #define MQMD_STRUC_ID "MD¬¬"
  #define MQFMT_STRING "MQSTR¬¬¬"
  ```

  Bu formu memcpy ve memcmp işlevleriyle kullanın.
- Diğer makro değişkeni değeri bir karakter dizisi olarak tanımlıyor; bu makro değişkeninin adı, _ARRAY ile birlikte suffixed dizgi biçiminin adıdır. Örneğin:

  ```
  #define MQMD_STRUC_ID_ARRAY 'M','D','¬','¬'
  #define MQFMT_STRING_ARRAY 'M','Q','S','T','R','¬','¬','¬'
  ```

  Bu formu, yapının bir eşgörünümü, MQMD_XX_ENCODE_CASE_ONE default makro değişkeni tarafından sağlanan değerlerden farklı değerlerle bildirildiğinde alanı kullanıma hazırlamak için kullanın.

Dinamik yapılar için ilk değerler

Bir yapının eşgörünümlerinin değişken sayısı gerektiğinde, yönetim ortamları tipik olarak, calloc ya da malloc işlevleri kullanılarak dinamik olarak elde edilen ana saklama alanı içinde yaratılır.

Bu tür yapılardaki alanları kullanıma hazırlamak için aşağıdaki teknik önerilir:

1. Yapıyı kullanıma hazırlamak için uygun MQxxx_XX_ENCODE_CASE_ONE default makro değişkenini kullanarak yapının bir eşgörünümünü bildirin. Bu yönetim ortamı, diğer yönetim ortamları için *model* olur:

   ```
   MQMD ModelMsgDesc = {MQMD_DEFAULT};
                                    /* declare model instance */
   ```

   Gerekli olduğu şekilde, model eşgörünümü statik ya da dinamik kullanım süresini vermek için bildirimde bulunan statik ya da otomatik anahtar sözcükleri kodlayın.
2. Yapıya ilişkin dinamik bir yönetim ortamı için depolama alanı elde etmek üzere calloc ya da malloc işlevlerini kullanın:

   ```
   PMQMD InstancePtr;
   InstancePtr = malloc(sizeof(MQMD));
                                    /* get storage for dynamic instance */
   ```
3. Model eşgörünümünü devingen eşgörünüme kopyalamak için memcpy işlevini kullanın:

   ```
   memcpy(InstancePtr,&ModelMsgDesc,sizeof(MQMD));
                                    /* initialize dynamic instance */
   ```

C++ dilinde kullan

C++ programlama dili için, üstbilgi kütükleri, yalnızca bir C++ derleyicisi kullanıldığında içerilen aşağıdaki ek deyimleri içerir:

```
#ifdef __cplusplus
  extern "C" {
#endif

/* rest of header file */

#ifdef __cplusplus
```

```
  }
#endif
```

COBOL içinde kodlama

COBOL ' de WebSphere MQ programları kodlanırken aşağıdaki bölümdeki bilgileri not edin.

Adlandırılmış Değişmezler

Değişmezlerin adları, adın bir parçası olarak altçizgi karakteri (_) içerir. COBOL ' de, tire karakterini (-) alt çizgi karakterini yerine kullanmanız gerekir. Karakter dizilimi değerlerine sahip sabit değerler, dizilim sınırlayıcısı olarak tek tırnak işareti (') karakterini kullanır. Derleyicinin bu karakteri kabul etmesini sağlamak için, APHOST derleyici seçeneğini kullanın.

CMQV kopya dosyası, adlandırılmış değişmezlerin bildirimlerini level-10 öğe olarak içerir. Sabitleri kullanmak için, level-01 öğesini açık bir şekilde bildirerek, değişmezlerin bildirimlerinde kopyalamak için COPY deyimini kullanın:

```
 WORKING-STORAGE SECTION.
 01  MQM-CONSTANTS.
     COPY CMQV.
```

Ancak bu yöntem, değişmezler, başvuruda bulunulmamış olsa da programdaki depolamayı işgal etmeye neden olur. Sabit değerler aynı çalıştırma birimindeki birçok ayrı programa dahil ediliyorsa, sabitlerin birden çok kopyası bulunur; bu, kullanılan ana saklama alanının önemli bir miktarına neden olabilir. You can avoid this by adding the GLOBAL clause to the level-01 declaration:

```
* Declare a global structure to hold the constants
 01  MQM-CONSTANTS GLOBAL.
     COPY CMQV.
```

Bu işlem, çalıştırma birimi içinde yalnızca *bir* sabit disk kümesi için depolama ayırır; ancak, sabitler, yalnızca level-01 bildirimini içeren program değil, çalıştırma birimindeki *herhangi bir* program tarafından başvurulabilir.

Yapı hizalamasının sağlanması

MQ çağrısının başlatılacak şekilde geçirilen IBM WebSphere MQ yapılarının sözcük sınırlarında hizalanması için özen gösterilebilir. 32 bitlik işlemler için 4 byte, 64 bit işlemleri için 8 bayt ve 128 bit işlemler için 16 bayt (IBM i) için 4 byte.

Mümkün olduğunda, tüm IBM WebSphere MQ yapılarını bir araya getirin, böylece tüm sınırları hizalanmış olur.

pTAL içinde kodlama

pTALiçinde IBM WebSphere MQ programlarını kodlarken aşağıdaki bölümdeki bilgileri not edin.

HP Integrity NonStop Server

IBM WebSphere MQ yapılarının tanımlanması ve kullanıma hazırlanması

IBM WebSphere MQ yapıları için pTAL yapı tanımlamaları ^DEFile biten adlarla verilir. Örneğin, aşağıdaki pTAL bildirimleri bir IBM WebSphere MQ Message Descriptor (MQMD) yapısı ve IBM WebSphere MQ Put Message Options (MQPMO) yapısı yaratmak için kodlanır.

```
    STRUCT MYMD(MQMD^DEF);        ! Declare an MQMD structure
    STRUCT MYPMO(MQPMO^DEF);      ! Declare an MQPMO structure
```

IBM WebSphere MQ , varsayılan değerlerle IBM WebSphere MQ yapılarını kullanıma hazırlamak için ^DEFAULT ile biten adlarla pTAL DEFINE sağlar. Bildirilen MQMD ve MQPMO yapılarına varsayılan değerler atamak için aşağıdaki pTAL deyimleri kodlanır:

```
MQMD^DEFAULT(MYMD);           ! Assign default values to an MQMD structure
MQPMO^DEFAULT(MYPMO);         ! Assign default values to an MQPMO structure
```

Benzer kodu kullanarak diğer IBM WebSphere MQ yapılarını bildirebilir ve kullanıma hazırlayabilirsiniz.

pTAL ve CRE

pTAL programları Ortak Çalıştırma Ortamı 'nı başlatamadığından, bu programlar C dili ya da COBOL ana yordamıyla kullanılmalıdır.

IBM WebSphere MQ ile sağlanan pTAL örnekleri, AMQSPTM0.C

MQCHAR veri tipi olan değiştirgeler

MQGET, MQPUT ve MQPUT1 yordamlarının her birinin `MQCHAR .EXT` veri tipi olan bir **`Buffer`** değiştirgesi vardır. Bu parametre, uygulamanın ileti verilerini göndermek ve almak için kullanılır.

Bu tür parametreler pTAL örneklerinde dizgi dizisi olarak gösterilir. Parametreleri bu şekilde bildirebilirsiniz, ancak genellikle bunları iletideki verilerin düzenini açıklayan yapı olarak bildirmeniz daha uygundur. Yordam değiştirgesi `MQCHAR .EXT`olarak bildirildi, ancak yordam çağrısında değiştirge olarak herhangi bir verinin adresi belirtilebilir.

Karakter dizilimlerini işleme

Kuyruk yöneticisi uygulamaya karakter verileri döndürdüğünde, kuyruk yöneticisi karakter verilerini her zaman alanın tanımlı uzunluğuna kadar boşluklarla doldurur. Kuyruk yöneticisi boş sonlandırılmış dizgiler döndürmez, ancak bunları girişinizde kullanabilirsiniz.

Visual Basic Kodlamasi

Visual Basic 'te WebSphere MQ programları kodlanırken aşağıdaki bölümdeki bilgileri not edin.

Not: .NET ortamının dışında, WebSphere MQ ' da Visual Basic (VB) desteği V6.0 düzeyinde sabitlendi. WebSphere MQ 7.0 ya da sonraki yayın düzeylerine eklenen en yeni işlev, VB uygulamaları için kullanılamaz. VB.NET, WebSphere MQ .NET sınıflarını kullanın. Daha fazla bilgi için bkz. .NET ' in kullanılması.

Visual Basic, yalnızca Windowsüzerinde desteklenir.

Visual Basic ile WebSphere MQarasında geçen ikili verilerin istenmeyen çevirisini önlemek için, MQSTRING yerine bir MQBYTE tanımlaması kullanın. CMQB.BAS , C byte tanımlamasına eşdeğer birkaç yeni MQBYTE tipi tanımlar ve bunları WebSphere MQ yapıları içinde kullanır. Örneğin, MQMD (ileti tanımlayıcı) yapısı için, MsgId (ileti tanıtıcısı) MQBYTE24olarak tanımlanır.

Visual Basic, bir işaretçi veri tipine sahip değildir, bu nedenle diğer WebSphere MQ veri yapılarına yapılan başvurular, işaretçi yerine göreli konuma göre olur. İki bileşen yapısından oluşan bir bileşik yapı bildirin ve çağrıya ilişkin bileşik yapıyı belirtin. WebSphere MQ Visual Basic için, bu işlemi mümkün kılmak için bir MQCONNXAny çağrısı sağlar ve istemci uygulamalarının istemci bağlantısında kanal özelliklerini belirtmesine olanak sağlar. Tipik MQCNO yapısının yerine tip atanmamış bir yapıyı (MQCNOCD) kabul eder.

MQCNOCD yapısı, MQCNO ve arkasından gelen bir MQCD ' den oluşan bir bileşik yapıdır. Bu yapı, CMQXB ' nin çıkış üstbilgi dosyası olarak bildirilmiş. MQCNOCD_DEFAULTS yordamını kullanarak bir MQCNOCD yapısını kullanıma hazırlayın. MQCONNX çağrıları yapan bir örnek (amqscnxb.vbp) sağlanıyor.

MQCONNX, MQCONNX ile aynı parametrelere sahiptir; ancak, *`ConnectOpts`* parametresinin MQCNO veri tipi yerine herhangi bir veri tipi olarak bildirilmiş olması dışında. Bu, işlevin MQCNO ya da MQCNOCD yapısını kabul etmesini sağlar. Bu işlev, ana üstbilgi dosyası CMQB ' de bildirilir.

IBM WebSphere MQ Nesne Modeli

IBM WebSphere MQ Nesne Modeli, sınıflardan, yöntemlerden ve özelliklerden oluşur. Bu bilgilerin her biri hakkında bilgi edinmek için bu bilgileri kullanın.

IBM WebSphere MQ Nesne Modeli aşağıdaki öğelerden oluşur:

- *Sınıflar* , kuyruk yöneticileri, kuyruklar ve iletiler gibi bilinen WebSphere MQ kavramlarını temsil eder.
- MQI çağrılarına karşılık gelen her sınıftaki*yöntemler* .
- WebSphere MQ nesnelerinin özniteliklerine karşılık gelen her sınıftaki*Özellikler* .

WebSphere MQ Nesne Modeli 'ni kullanarak bir WebSphere MQ uygulaması yaratırken, programda bu sınıfların somut örnekleri yaratıyorsunuz. Nesne yönelimli programlamada bir sınıfın somut örneğinin adı *nesne*olarak adlandırılır. Bir nesne yaratıldığında, nesnenin özelliklerinin değerlerini inceleyerek ya da ayarlarken (MQINQ ya da MQSET çağrısının verilmesinin eşdeğeri) ve nesne için yöntem çağrıları yaparak (diğer MQI çağrılarının verilmesiyle eşdeğeri) nesneyle etkileşimde bulunabilrsiniz.

Bu konular, ayrıntılı olarak WebSphere MQ Nesne Modelleri 'nin her birini açıklar:

- “Sınıflar” sayfa 84
- “nesne başvuruları” sayfa 85
- “Dönüş kodları” sayfa 85

Sınıflar

WebSphere MQ Object Model, aşağıdaki temel sınıf kümesini sağlar.

Modelin gerçek uygulaması, desteklenen farklı nesne yönelimli ortamlar arasında biraz değişiklik gösterir.

MQQueueManager
MQQueueManager sınıfının bir nesnesi, kuyruk yöneticisine yönelik bir bağlantıyı temsil eder. Connect (), Disconnect (), Commit () ve Backout () için yöntemleri vardır (MQCONN ya da MQCONNX, MQDISC, MQCMIT ve MQBACK ' ın eşdeğeridir). Bir kuyruk yöneticisinin özniteliklerine karşılık gelen özellikleri vardır. Bir kuyruk yöneticisi özniteliği özelliğine erişilmesi, önceden bağlantı kurulmamışsa kuyruk yöneticisine örtük olarak bağlanır. Bir MQQueueManager nesnesinin yok olması kuyruk yöneticisinden örtük olarak bağlantıyı keser.

MQQueue
MQQueue sınıfından bir nesne bir kuyruğu gösterir. Kuyruktan ve kuyruktan (MQPUT ve MQGET ile eşdeğer) iletiler () ve Get () için yöntemler içerir. Bir kuyruğun özniteliklerine karşılık gelen özellikleri vardır. Bir kuyruk özniteliği özelliğine erişilmesi ya da bir put () ya da get () yöntemi çağrısının verilmesi, kuyruğu örtük olarak açar (bu MQPEL ' in eşdeğeridir). Bir MQQueue nesnesini yok etmek, kuyruğu örtük olarak kapar (MQCLOSE ' nin eşdeğeri).

MQTopic
MQTopic sınıfından bir nesne bir konuyu gösterir. Bu konuya (MQPUT ve MQGET eşdeğeri) ilişkin iletiler () (yayınlama) ve Get () (alma ya da abone olma) iletileri yerleştirme (alma ya da abone olma) yöntemlerine sahiptir. Bir konunun özniteliklerine karşılık gelen özellikleri vardır. Bir MQTopic nesnesine yalnızca yayın ya da abonelik için erişilebilir, aynı anda hem eşzamanlı olarak değil hem de abonelik için erişilebilir. İleti almak için kullanıldığında, MQTopic nesnesi yönetilmeyen ya da yönetilen bir abonelikle ve dayanıklı ya da dayanıklı olmayan bir abone olarak yaratılabilir. Bu farklı senaryolar için çok sayıda aşırı yüklenmiş oluşturucular sağlanır.

MQMessage
MQMessage sınıfının bir nesnesi, kuyruğa konmak üzere bir iletiyi gösterir ya da kuyruktan alındı. Bir arabellek içerir ve hem uygulama verilerini, hem de MQMD ' yi sarmalayın. Bu, MQMD alanlarına karşılık gelen özellikleri ve (örneğin, dizgiler, uzun tamsayılar, kısa tamsayılar, tek baytlar) arabelleğinden ve arabellekten kullanıcı verilerini yazmanızı ve okumanızı sağlayan yöntemlere sahiptir.

MQPutMessageSeçenekleri
MQPutMessageSeçenekleri sınıfından bir nesne, MQPMO yapısını gösterir. Bu, MQPMO alanlarına karşılık gelen özelliklere sahiptir.

MQGetMessageSeçenekleri
MQGetMessageSeçenekleri sınıfından bir nesne MQGMO yapısını gösterir. Bu, MQGMO alanlarına karşılık gelen özelliklere sahiptir.

MQProcess
MQProcess sınıfının bir nesnesi bir süreç tanımlamasını gösterir (tetikleme ile kullanılır). Bir süreç tanımlamasının özniteliklerini temsil eden özelliklere sahiptir.

MQDistributionList
MQDistributionList sınıfının bir nesnesi bir dağıtım listesini gösterir (tek bir MQPUT ile birden çok ileti göndermek için kullanılır). MQDistributionListÖğe nesnelerinin bir listesini içerir.

MQDistributionListÖğesi
MQDistributionListÖğe sınıfının bir nesnesi tek bir dağıtım listesi hedefini gösterir. MQOR, MQRR ve MQPMR yapılarını sarsalır ve bu yapıların alanlarına karşılık gelen özellikleri içerir.

nesne başvuruları

MQI kullanan bir WebSphere MQ programında, WebSphere MQ , program için bağlantı noktaları ve nesne tanıtıcılarını döndürür.

Bu tutamaçlar, sonraki WebSphere MQ çağrılarında parametre olarak geçirilmelidir. WebSphere MQ Nesne Modeli ile bu tutamaçlar, uygulama programından gizlenir. Bunun yerine, bir nesne başvurusundan uygulama programına döndürülemekte olan bir sınıf sonuçlarından bir nesne yaratılması. Nesne için yöntem çağrıları ve özellik erişimleri yapılırken kullanılan bu nesne başvurudur.

Dönüş kodları

Bir yöntem çağrısının verilmesi ya da bir özellik değerinin ayarlanması, dönüş kodlarının belirlenmesine neden olur.

Bu dönüş kodları bir tamamlama kodudur ve bir neden kodudur ve nesnenin kendi özellikleridir. Tamamlanma kodu ve neden kodunun değerleri, nesne yönelimli ortama özgü bazı ek değerler ile, MQI için tanımlananlarla aynıdır.

JMS için Java ya da IBM WebSphere MQ sınıfları için IBM WebSphere MQ sınıfları kullanmalı mıyım?

Bir Java uygulaması, JMS için IBM WebSphere MQ sınıflarını ya da JMS için IBM WebSphere MQ sınıflarını kullanarak IBM WebSphere MQ kaynaklarına erişebilirler. Her yaklaşımın avantajları vardır.

Java içinIBM WebSphere MQ sınıfları, Message Queue Interface (Message Queue Interface; İleti Kuyruğu Arabirimi), yerel IBM WebSphere MQ API 'sını içerir ve diğer nesne yönelimli arabirimlerle aynı nesne modelini kullanır; oysa, Java Message Service için IBM WebSphere MQ sınıfları, Sun 'ın Java Message Service (JMS) arabirimlerini uygular.

Java dışındaki ortamlardan IBM WebSphere MQ ' e alışıksanız, yordamsal ya da nesne yönelimli diller kullanarak, Java için IBM WebSphere MQ sınıflarını kullanarak var olan bilginizi Java ortamına aktarabilirsiniz. Ayrıca, JMS için IBM WebSphere MQ sınıflarında bulunan tüm IBM WebSphere MQözelliklerinde de tüm özellik yelpazesinde kullanılabilir.

IBM WebSphere MQile ilgili bilgi sahibi değilseniz ya da JMS deneyiminiz varsa, JMS için IBM WebSphere MQ sınıflarını kullanarak, IBM WebSphere MQ kaynaklarına erişmek için bilinen JMS API ' yı kullanmayı daha kolay bulabilirsiniz. JMS, aynı zamanda Java Platform, Enterprise Edition (Java EE) platformunun ayrılmaz bir parçasıdır. Java EE uygulamaları, iletileri zamanuyumsuz olarak işlemek için ileti odaklı Bean 'leri (MDBs) kullanabilir ve MDBs yalnızca JMS iletilerini işleyebilir. JMS ayrıca, IBM WebSphere MQgibi zamanuyumsuz ileti sistemi sistemleriyle etkileşimde bulunmak için Java EE için standart bir mekanizmadır. Java EE ile uyumlu olan her uygulama sunucusu bir JMS sağlayıcısı içermelidir; bu nedenle, farklı uygulama sunucuları arasında iletişim kurmak için JMS ' yi kullanabilir ya da uygulamada herhangi bir değişiklik yapmadan bir uygulamayı bir JMS sağlayıcısından başka bir sunucuya çevirebilirsiniz.

IBM WebSphere MQ uygulamalarını tasarlama

Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

Bir IBM WebSphere MQ uygulaması tasarlarken aşağıdaki sorular ve seçenekler göz önünde bulundurulr:

Uygulama tipi

Uygulamanızın amacı nedir? Geliştirebileceğiniz farklı uygulama tipleriyle ilgili bilgi edinmek için aşağıdaki bağlantılara bakın:

- Sunucu
- İstemci
- Yayınla/abone ol
- Web hizmetleri
- Kullanıcı çıkışları, API çıkışları ve kurulabilir hizmetler

Buna ek olarak, IBM WebSphere MQyönetimini otomatikleştirmek için kendi uygulamalarınızı da yazabilirsiniz. Daha fazla bilgi için bkz. WebSphere MQ Administration Interface (MQAI) ve Automated admining tasks (Yönetim görevlerinin otomatikleştirilmesi) konusuna giriş.

Programlama dili

IBM WebSphere MQ , uygulama yazmak için bir dizi yordamsal ve nesne yönelimli programlama diline destek sağlar. Daha fazla bilgi için bkz. “Kullanılacak programlama diline karar verme” sayfa 75.

Birden fazla platform için uygulamalar

Uygulamanız birden çok platformda çalıştırılır mı? bugün kullandığınız bir platformdan farklı bir platforma taşınmak için bir stratejiniz var mı? Bu sorulardan herhangi birinin yanıtı evet ise, programlarınızı platform bağımsızlığı için kodladığınızdan emin olun.

C kullanıyorsanız, kod ANSI standart C ' de kodlanır. Platforma özgü işlev daha hızlı ya da daha verimli olsa bile, eşdeğer bir platforma özgü işlev yerine standart bir C kitaplığı işlevi kullanın. Bu kural dışı durum, #ifdefkullanılarak her iki durum için kodladığınızda, koddaki verimliliğin ayrıştırılması durumunda olur. Örneğin:

```
#ifdef _AIX
     AIX specific code
#else
     generic code
#endif
```

Kuyruk tipleri

Her gereksinim duyarsanız bir kuyruk yaratmak istiyor musunuz, yoksa önceden ayarlanmış kuyrukları kullanmak mı istiyorsunuz? Kullanmayı bitirdiğinizde bir kuyruğu silmek istiyor musunuz, yoksa yeniden kullanılacak bir kuyruk mu? Uygulama bağımsızlığı için diğer ad kuyruklarını kullanmak istiyor musunuz? Hangi kuyrukların desteklendiğini görmek için Kuyruklar konusuna bakın.

Kuyruk yöneticisi kümelerini kullanma

Kümeleri kullanırken mümkün olan basitleştirilmiş sistem yönetiminden ve artırılabilir düzeyde kullanılabilirlik, ölçeklenebilirlik ve iş yükü dengelemesi avantajlarından yararlanmak isteyebilirsiniz. Ek bilgi için Kuyruk yöneticisi kümeleri başlıklı konuya bakın.

İleti tipleri

Basit iletiler için veri paketleri kullanmak isteyebilirsiniz, ancak diğer durumlar için istek iletileri (yanıt beklediğiniz) istemeniz gerekebilir. İletilerinizin bazılarına farklı öncelikler atamak isteyebilirsiniz. İleti tasarlanmasına ilişkin daha fazla bilgi için bkz. “İletilerinizi tasarlama” sayfa 88.

Yayınlama/abone olma ya da noktadan noktaya ileti alışverişi kullanma

Yayınlama/abone olma ileti alışverişi kullanılarak, bir gönderme uygulaması, IBM WebSphere MQ iletisinde paylaşmak istediği bilgileri IBM WebSphere MQ yayınla yönetilen standart bir hedefe gönderiyor? Abone olun ve IBM WebSphere MQ ' in bu bilgilerin dağıtımını işlemesini sağlar. Hedef uygulamanın aldığı bilgilerin kaynağı hakkında herhangi bir bilgi sahibi olması gerekmez, yalnızca bir

ya da daha fazla konuya ilgi kaydeder ve bu bilgileri kullanılabilir olduğunda alır. Yayınlama/abone olma ileti alışverişi hakkında daha fazla bilgi için bakınız: Introduction to IBM WebSphere MQ publish/ subscreen Messaging .

Bir gönderme uygulaması, noktadan noktaya ileti sistemini kullanarak, belirli bir kuyruğa ileti gönderir. Bu ileti, bir alma uygulamasının onu alabileceği bir kuyruğa gönderir. Alma uygulaması, belirli bir kuyruktan iletileri alır ve içerikleri üzerinde işlem yapar. Uygulama genellikle hem gönderici, hem de alıcı olarak işlev görecektir, başka bir uygulamaya sorgu gönderip yanıt alıcaktır.

IBM WebSphere MQ programlarınızı denetleme

Bazı programları otomatik olarak başlatmak ya da belirli bir ileti kuyruğa gelene kadar (IBM WebSphere MQ *tetikleme* özelliğini kullanarak) program beklemeniz gerekebilir. Bkz. “Starting IBM WebSphere MQ applications using triggers” sayfa 313). Diğer bir seçenek olarak, bir kuyrukta bulunan iletiler yeterince hızlı işlenmediklerinde, uygulamanın başka bir eşgörünümünü başlatmak isteyebilirsiniz (Özel işlemden geçirme olaylarıiçinde açıklandığı gibi IBM WebSphere MQ *izleme kodu ekleme olayları* özelliğini kullanarak).

Uygulamanızın bir IBM WebSphere MQ istemcisinde çalıştırılması

İstemci ortamında tam MQI desteklenir ve bu, neredeyse herhangi bir IBM WebSphere MQ uygulamasının bir IBM WebSphere MQ MQI istemcisinde çalışmak üzere yeniden bağlanmasına olanak sağlar. Link the application on the IBM WebSphere MQ MQI client to the MQIC library, rather than to the MQI library.

Not: Bir IBM WebSphere MQ istemcisinde çalışan bir uygulama, koşutzamanlı olarak birden çok kuyruk yöneticisine bağlanabilir ya da bir MQCONN ya da MQCONNX çağrısında yıldız (*) ile bir kuyruk yöneticisi adı kullanabilir. İstemci kitaplıkları yerine kuyruk yöneticisi kitaplıklarına bağlantı oluşturmak istiyorsanız, bu işlev kullanılamayacağını uygulamayı değiştirin.

Ek bilgi için “Uygulamaların IBM WebSphere MQ MQI istemci ortamında çalıştırılması” sayfa 342 başlıklı konuya bakın.

Uygulama performansı

Design decisions can impact your application performance, for suggestions for enhancing performance of IBM WebSphere MQ applications, see “Uygulama tasarımı ve performansı” sayfa 89.

Gelişmiş IBM WebSphere MQ teknikleri

Daha ileri düzey uygulamalar için, yanıtları ilintilendirmek ve IBM WebSphere MQ bağlam bilgilerini oluşturmak ve göndermek gibi bazı gelişmiş IBM WebSphere MQ tekniklerini kullanmak isteyebilirsiniz. Daha fazla bilgi için bkz “Gelişmiş IBM WebSphere MQ teknikleri” sayfa 90.

Verilerinizin güvenliğini sağlama ve bütünlüğünün korunması

İletinin kabul edilebilir bir kaynaktan gönderildiğini test etmek için iletiyle geçirilen bağlam bilgilerini kullanabilirsiniz. Verilerinizin diğer kaynaklarla tutarlı olmasını sağlamak için IBM WebSphere MQ 'in ya da işletim sisteminizin sağladığı uyumluluk olanaklarını kullanabilirsiniz (ek ayrıntılar için “İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308 ' a bakın). Önemli iletilerin sunulmasını sağlamak için IBM WebSphere MQ iletilerinin *persistence* özelliğini kullanabilirsiniz.

IBM WebSphere MQ uygulamalarının test edilmesi

IBM WebSphere MQ programları için uygulama geliştirme ortamı, başka bir uygulama için bundan farklı değildir; bu nedenle, IBM WebSphere MQ izleme olanaklarının yanı sıra aynı geliştirme araçlarını da kullanabilirsiniz.

Kural dışı durumlar ve hatalar işleniyor

Teslim edilemeyen iletilerin nasıl işleneceğini ve kuyruk yöneticisi tarafından size bildirilen hata durumlarının nasıl çözümleneceğini göz önünde bulundurmanız gerekir. Bazı raporlar için, rapor seçeneklerini MQPUT üzerine ayarlamanız gerekir.

İlgili kavramlar

IBM WebSphere MQ teknik genel bilgileri

“Uygulama geliştirme kavramları” sayfa 7

IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

“Kuyruğa alma uygulaması yazılıyor” sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

“İstemci uygulamaları yazılıyor” sayfa 335
What you need to know to write client applications on WebSphere MQ.

“.NET kullanımı” sayfa 537
.NET için WebSphere MQ sınıfları, .NET programlama çerçevesinde yazılmış bir programın WebSphere MQ ' a bir WebSphere MQ MQI istemcisi olarak bağlanmasını ya da doğrudan bir WebSphere MQ Server sunucusuna bağlanmasını sağlar.

“C++ kullanılması” sayfa 603
WebSphere MQ provides C++ classes equivalent to WebSphere MQ objects and some additional classes equivalent to the array data types. Bu, MQI aracılığıyla olmayan bazı özellikleri sağlar.

“JMS için WebSphere MQ sınıflarının kullanılması” sayfa 687
Java Message Service için WebSphere MQ sınıfları (JMS için WebSphere MQ sınıfları), WebSphere MQile birlikte verilen JMS sağlayıcısıdır. javax.jms paketinde tanımlanan arabirimlerin yanı sıra, JMS için WebSphere MQ sınıfları JMS API ' ya iki uzantı kümesi sağlar.

“Java için WebSphere MQ sınıflarının kullanılması” sayfa 627
Java için WebSphere MQ sınıfları, Java ortamında WebSphere MQ ' yı kullanmanıza olanak sağlar. A Java application can use either WebSphere MQ classes for Java or WebSphere MQ classes for JMS to access WebSphere MQ resources.

“Component Object Model Interface olanağının kullanılması (ActiveXiçin WebSphere MQ Automation Sınıfları)” sayfa 992
WebSphere MQ Automation Classes for ActiveX (MQAX), uygulamanıza WebSphere MQ' a erişmek için kullanabileceğiniz sınıfları sağlayan ActiveX bileşenleridir.

İletilerinizi tasarlama

Bu bilgiler içinde verilen bilgileri, iletileri tasarlamaya yardımcı olmak için göz önünde bulundurun.

İletiyi kuyruğa koymak için bir MQI çağrısı kullandığınızda bir ileti oluşturursiniz. Aramaya giriş olarak, bir *ileti tanımlayıcısı* (MQMD) ve başka bir programa göndermek istediğiniz veriler için bazı denetim bilgileri sağlanmanız gerekir. Ama tasarım aşamasında, aşağıdakileri göz önünde bulundurmanız gerekir, çünkü onlar sizin mesajlarınızı nasıl oluşturmanızı etkiler:

Kullanılacak ileti tipi
Bir ileti gönderebileceğiniz basit bir uygulama tasarlıyor musunuz, daha sonra başka bir işlem yapmak zorunda kalmadınız mı? Yoksa bir soruyla cevap mı istiyorsun? Bir soru soruyorsanız, iletiyi almak istediğiniz kuyruğun adını ileti tanımlayıcısına dahil edebilirsiniz.

İsteğinizin ve yanıt iletilerinin zamanuyumlu olmasını istiyor musunuz? Bu, yanıtta yanıtlamak için bir zamanaşımı süresi ayarladığınızı belirtir; yanıtı bu süre içinde almezseniz, yanıt bir hata olarak işlenir.

Ya da işlemlerinizin ortak zamanlama sinyalleri gibi belirli olayların ortaya çıkmalarına bağlı kalmaması için zamanuyumsuz olarak çalışmayı tercih eder misiniz?

Diğer bir dikkat, tüm mesajlarınızın bir çalışma birimi içinde olup olmamadır.

İletilere farklı öncelikler atama
Her iletiye bir öncelik değeri atayabilir ve kuyruk tanımlayabilir ve böylece, iletilerin öncelikleri sırasına göre iletileceğini belirler. Bunu yapmazsanız, başka bir program kuyruktan bir ileti aldıklarında, her zaman en yüksek önceliğe sahip iletiyi alır. Kuyruk, iletilerini öncelik sırasına göre korumuyorsa, kuyruktan ileti alan bir program, iletileri kuyruğa eklendikleri sırayla alır.

Ayrıca, iletiler kuyruğa konduğunda kuyruk yöneticisinin atadığı tanıtıcıyı kullanarak bir ileti de seçebilir. Diğer bir seçenek olarak, iletilerinizin her biri için kendi tanıtıcılarınızı da oluşturabilirsiniz.

Kuyruklardaki kuyruk yöneticisinin yeniden başlatılmasına etkisi
The queue manager preserves all persistent messages, recovering them when necessary from the WebSphere MQ log files, when it is restarted. Kalıcı olmayan iletiler ve geçici dinamik kuyruklar

korunmaz. Atılmamasını istemediğiniz iletiler, yaratıldıklarında kalıcı olarak tanımlanmalıdır. UNIX and Linux sistemlerinde WebSphere MQ for Windows ya da WebSphere MQ için bir uygulama yazarken, günlük dosyası sınırlamalarına çalışacak bir uygulama tasarlama riskini azaltmak için, sisteminizin günlük dosyası ayırma açısından nasıl ayarlandığını bildiğinizden emin olun.

İletilerin alıcısına kendinizle ilgili bilgiler verme
Genellikle, kuyruk yöneticisi kullanıcı kimliğini ayarlar, ancak uygun şekilde yetkili uygulamalar bu alanı da ayarlayabilir; böylece, kendi kullanıcı kimliğinizi ve alma programının hesap ya da güvenlik amacıyla kullanabileceği diğer bilgileri de ekleyebilirsiniz.

Alma kuyruklarının miktarı
Bir iletinin birkaç kuyruğa konması gerekiyorsa, bir dağıtım listesikullanabilir ya da bir konu için yayınlayabilirsiniz.

Uygulama tasarımı ve performansı

Düşük program tasarımının performansı etkileyebileceği bir dizi yol vardır. Bu durum, programın kendisini iyi bir şekilde gerçekleştirebileceği, ancak diğer görevlerin başarımını olumsuz yönde etkileyebileceği için algılaması zor olabilir. Bu konuda, WebSphere MQ çağrıları yapan programlara özgü bazı sorunlar açıklanmaktadır.

Verimli uygulamaları tasarlamanıza yardımcı olacak birkaç fikir aşağıda bulunur:

- Uygulamanızı, bir kullanıcının düşünme süresiyle paralel olarak işlenmek üzere tasarlayın:
  - Bir pano görüntüleyin ve uygulamanın başlatılmasına devam ederken kullanıcının yazmaya başlamasını sağlayın.
  - Gereksinim duyardığınız verileri farklı sunuculardan elde edin.
- Sürekli olarak açma ve kapama, bağlama ve bağlantı kesme yerine bunları yeniden kullanacaksanız, bağlantıların ve kuyrukların açık kalmasını sağlar.
- Ancak, yalnızca bir ileti yerleştiren bir sunucu uygulaması MQPUT1' i kullanmalıdır.
- Kuyruk yöneticileri, 4 KB ile 100 KB arasındaki iletiler için eniyilenir. Çok büyük iletiler verimsizdir; tek bir 100 MB 'lik iletiden her biri 1 MB' lik 100 ileti göndermek daha iyi olur. Çok küçük mesajlar da verimsiz. Kuyruk yöneticisi, 4 KB ' lik bir iletide olduğu gibi tek byte 'lık bir ileti için de aynı iş miktarını gerçekleştirir.
- İletilerinizi bir çalışma birimi içinde tutun; böylece, bu iletiler eşzamanlı olarak kesinleştirilebilir ya da yedeklenebilirler.
- Kurtarılabilir olması gerekmeyen iletiler için kalıcı olmayan seçeneği kullanın.
- Bir iletiyi hedef kuyruklara göndermeniz gerekiyorsa, bir dağıtım listesi kullanmayı düşünün.

İleti uzunluğunun etkisi

Bir iletteki veri miktarı, iletiyi işleyen uygulamanın performansını etkileyebilir. Uygulamanızın en iyi performansını elde etmek için yalnızca bir iletiyle temel verileri gönderin. Örneğin, bir banka hesabını borç istemek için, istemciden sunucu uygulamasına geçilmesi gerekebilecek tek bilgi, hesap numarası ve borç tutarının olduğu anlamına gelir.

İleti kalıcılığın etkisi

Kalıcı iletiler genellikle günlüğe kaydedilir. Günlüğe kaydetme iletileri, uygulamanızın performansını azaltır, bu nedenle yalnızca temel veriler için kalıcı iletiler kullanın. Kuyruk yöneticisi durdurursa ya da başarısız olursa, bir iletteki veriler atılabilir ise, kalıcı olmayan bir ileti kullanın.

Belirli bir iletiyi arama

MQGET çağrısı genellikle bir kuyruktan ilk iletiyi alır. Belirli bir iletiyi belirtmek için ileti tanımlayıcısındaki ileti ve ilinti tanıtıcılarını (*MsgId* ve *CorrelId*) kullanırsanız, kuyruk yöneticisi bu iletiyi buluncaya kadar

kuyrukta arama yapmak zorundadır. MQGET çağrısının bu şekilde kullanılması, uygulamanızın başarımını etkiler.

Farklı uzunluklara ilişkin iletileri içeren kuyruklar

Uygulamanız sabit bir uzunluğa ilişkin iletileri kullanamıyorsa, arabellekleri büyüyüp küçültmeyi dinamik olarak tipik ileti boyutuna uygun olarak küçültür. Uygulama, arabellek çok küçük olduğu için başarısız olan bir MQGET çağrısını yayınlarsa, ileti verilerinin boyutu döndürülür. Arabelleğin uygun şekilde yeniden boyutlandırılması ve MQGET çağrısının yeniden yayınlanması için uygulamanızı kod ekleyin.

Not: *MaxMsgLength* özniteliğini açık bir şekilde ayarlamadıysanız, varsayılan değer 4 MB ' dir ve bu, uygulama arabelleği boyutunu etkilemek için kullanılırsa çok verimsiz olabilir.

Eşitleme noktalarının sıklığı

Eşitleme noktası içinde çok büyük sayıda MQPUT ya da MQGET komutu veren programlar, bunları kesinleştirmeden, performans sorunlarına neden olabilir. Etkilenen kuyruklar, şu anda erişilemez olan iletilerle veri doldurabilir; diğer görevler bu iletileri almak için bekliyor olabilir. Bunun, depolama alanı açısından ve ileti almaya çalışan görevlerle bağlantılı iş parçacıkları açısından etkileri vardır.

MQPUT1 çağrısının kullanılması

Use the MQPUT1 call only if you have a single message to put on a queue. Birden çok ileti koymak istiyorsanız, MQOPEN çağrısını kullanın ve ardından bir dizi MQPUT çağrısını ve tek bir MQCLOSE çağrısını kullanın.

Kullanıdaki iş parçacıklarının sayısı

For WebSphere MQ for Pencereler, an application might require a large number of threads. Her kuyruk yöneticisi işlemi, izin verilen uygulama iş parçacığı sayısı üst sınırı olarak ayrılır.

Uygulamalar çok fazla iş parçacığı kullanabilir. Uygulamanın bu olasılığı dikkate alıp almadığını ve bu tür oluş tipini raporlamak veya raporlamak için gerekli önlemleri aldığını göz önünde bulundurun.

Gelişmiş IBM WebSphere MQ teknikleri

Basit bir IBM WebSphere MQ uygulaması için, uygulamanıza hangi WebSphere MQ nesnelerine ve hangi ileti türlerini kullanmak istediğinize karar vermeniz gerekir. Daha gelişmiş bir uygulama için aşağıdaki bölümlerde tanıtılan tekniklerden bazılarını kullanmak isteyebilirsiniz.

İleti bekleniyor

Bir kuyruğa hizmet veren bir program, aşağıdaki işlemi yaparak iletileri bekleyebilirler:

- Bir ileti gelene kadar bekleme ya da belirtilen bir zaman aralığı kullanım süresi doluyor (bkz. “İleti bekleniyor” sayfa 256).
- Bir ileti geldiğinde yönlendirilecek bir geri bildirme çıkışı oluşturma; bkz. “IBM WebSphere MQ iletilerinin zamanuyumsuz tüketimi” sayfa 32.
- Bir iletinin geldiğini görmek için kuyruğun düzenli olarak çağrılması (*yoklama*). Bu, tipik olarak önerilmez, çünkü performans sonuçları olabilir.

Yanıt ilintilendirme

WebSphere MQ uygulamalarında, program bazı işleri yapmak için istekte bulunan bir ileti aldığında, program istekçiye genellikle bir ya da daha fazla yanıt iletisi gönderir.

İstekte bulunanın bu yanıtları özgün isteğiyle ilişkilendirmesine yardımcı olmak için, bir uygulama her iletinin tanımlayıcısında bir *ilinti tanıtıcısı* alanı ayarlayabilir. Programlar, istek iletisinin ileti tanıtıcısını, yanıt iletilerinin ilinti tanıtıcısı alanına kopyalar.

Bağlam bilgilerinin ayarlanması ve kullanılması

Bağlam bilgileri , iletileri oluşturan kullanıcıyla iletileri ilişkilendirmek için ve iletiyi oluşturan uygulamayı tanımlamak için kullanılır. Bu tür bilgiler, güvenlik, muhasebe, denetim ve sorun belirleme için kullanışlıdır.

Bir ileti yarattığınızda, kuyruk yöneticisinin varsayılan bağlam bilgilerini iletinizle ilişkilendirmesini isteyen bir seçenek belirleyebilirsiniz.

Bağlam bilgilerini kullanma ve ayarlama hakkında daha fazla bilgi için bkz. “İleti bağlamı” sayfa 37.

WebSphere MQ programlarının otomatik olarak başlatılması

Bir kuyruğa ileti geldiğinde otomatik olarak bir program başlatmak için WebSphere MQ *tetikleme* öğesini kullanın.

Bir programın bu kuyruğu işlemeye başlaması için, tetikleme koşullarını bir kuyruğun üzerinde ayarlayabilirsiniz:

- Kuyruğa her ileti geldiğinde
- Kuyruğa ilk ileti geldiğinde
- Kuyruklardaki ileti sayısı önceden tanımlanmış bir sayıya ulaştığında

Tetikleme hakkında daha fazla bilgi için bkz. “Starting IBM WebSphere MQ applications using triggers” sayfa 313. Tetikleme, bir programı otomatik olarak başlatmanın tek bir yolu. Örneğin,WebSphere MQ dışı olanakları kullanan bir süreölçerin üzerinde otomatik olarak bir program başlatabilirsiniz.

WebSphere MQ , kuyruk yöneticisi başlatıldığında WebSphere MQ programlarının başlatılacağı hizmet nesnelerini tanımlayabilir; bkz. Hizmet nesneleri.

WebSphere MQ raporlarının oluşturulması

Bir uygulama içinde aşağıdaki raporları isteyebilirsiniz:

- Özel durum raporları
- Süre bitimi raporları
- Varışta doğrulama (COA) raporları
- Teslim edilme (COD) raporları
- Pozitif işlem bildirimi (PAN) raporları
- Negatif işlem bildirimi (NAN) raporları

Bunlar “Rapor iletileri” sayfa 11içinde açıklanmaktadır.

Kümeler ve ileti zenginlikleri

Aynı kuyruk için birden çok tanımı olan kümeleri kullanmaya başlamadan önce, ilgili ileti alışverişi gerektiren herhangi bir uygulama olup olmadığını görmek için uygulamalarınızı inceleyin.

Bir küme içinde, bir ileti, uygun kuyruğun bir örneğini barındıran herhangi bir kuyruk yöneticisine yönlendirilebilir. Bu nedenle, ileti zenginlikleri olan uygulamaların mantığı bozunabilir.

Örneğin, soru ve yanıt formlarında aralarında akan bir dizi ileti dizisine dayanan iki uygulamanız olabilir. Tüm soruların aynı kuyruk yöneticisine gönderilmesi ve tüm yanıtların diğer kuyruk yöneticisine geri gönderildiği önemli olabilir. Bu durumda, iş yükü yönetimi yordamında, iletileri uygun kuyruğun bir eşgörünümüne ev sahipliği yapmak üzere olan herhangi bir kuyruk yöneticisine göndermemesi önemlidir.

Mümkün olan yerlerde, kötüleri ortadan kaldırın. İleti zenginliğinin kaldırılması, uygulamaların kullanılabilirliğini ve ölçeklenebilirliğini artırır.

Daha fazla bilgi için bakınız: Handling message affinities .

Örnek WebSphere MQ programları

Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

- “Dağıtılmış platformlar için örnek programlar” sayfa 92

İlgili kavramlar

“Uygulama geliştirme kavramları” sayfa 7
IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“IBM WebSphere MQ uygulamalarını tasarlama” sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

“Kuyruğa alma uygulaması yazılıyor” sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

“İstemci uygulamaları yazılıyor” sayfa 335
What you need to know to write client applications on WebSphere MQ.

“ WebSphere MQ' da Web hizmetlerini kullanma” sayfa 907
SOAP için IBM WebSphere MQ iletimi ya da HTTP için IBM WebSphere MQ köprüsü kullanılarak Web hizmetleri için IBM WebSphere MQ uygulamaları geliştirebilirsiniz.

“Yayınlama/abone olma uygulamaları yazılıyor” sayfa 266
Yayınlama/abone olma WebSphere MQ uygulamalarını yazmaya başlayın.

“IBM WebSphere MQ uygulaması oluşturulması” sayfa 409
Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

“Program hatalarının işlenmesi” sayfa 526
Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

Dağıtılmış platformlar için örnek programlar

Bu konuda, C ve COBOL yazılımında yazılmış, IBM WebSphere MQile verilen örnek programlar açıklanmaktadır. Örnekler, Message Queue Interface (MQI) ' ın tipik kullanımları gösterir.

Örnekler, genel programlama tekniklerini göstermek üzere tasarlanmadığından, bir üretim programına dahil etmek isteyebileceğiniz bazı hata denetimi atlanır. Ancak, bu örnekler kendi ileti kuyruklama programlarınız için temel olarak kullanıma uygundur.

Tüm örneklere ilişkin kaynak kodu ürünle birlikte sağlanır; bu kaynak, programlarda gösterilen ileti kuyruklama tekniklerini açıklayan yorumları içerir.

C++ örnek programları: C + + içinde bulunan örnek programların bir açıklaması için bkz. C++ kullanılması .

Örneklerin adları amqönekiyle başlar. Dördüncü karakter, programlama dilini ve derleyiciyi gerekli yerlerde gösterir.

s	C dili
0	Hem IBM hem de Micro Focus derleyicilerindeki COBOL dili
i	Yalnızca IBM derleyicilerindeki COBOL dili

m COBOL dili yalnızca Micro Focus derleyicileri üzerinde

Yürütülebilir dosyanın sekizinci karakteri, örneğin yerel bağlama kipinde mi, yoksa istemci kipinde mi çalıştırılacağını gösterir. Sekizinci karakter yoksa, örnek yerel bağ tanımları kipinde çalıştırılır. Sekizinci karakter ' c' ise, örnek istemci kipinde çalıştırılır. Kuyruk yöneticisini istemci bağlantılarını kabul etmek üzere ayarlamak için, ayrıntılar için bkz. “Örnek programların hazırlanması ve çalıştırılması” sayfa 104 .

Örnek programlar hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “Örnek programlar içinde gösterilen özellikler” sayfa 93
- “Yayınlama/Abone Olma örnek programları” sayfa 129
- “Put Sample programlar” sayfa 134
- “Dağıtım Listesi örnek programı” sayfa 122
- “Göz At örnek programları” sayfa 110
- “Tarayıcı örnek programı” sayfa 111
- “Get Sample programlar” sayfa 123
- “Reference Message Sample programlar” sayfa 134
- “İstek örnek programları” sayfa 140
- “Sorma örnek programları” sayfa 128
- “Message Handle örnek programının Sorgusu Özellikleri” sayfa 129
- “Örnek programları ayarla” sayfa 144
- “Echo örnek programları” sayfa 122
- “Data-Conversion örnek programı” sayfa 114
- “Tetikleme örnek programları” sayfa 147
- “Zamanuyumsuz Koy örnek programı” sayfa 109
- “Veritabanı koordinasyonu örnekleri” sayfa 114
- “CICS hareket örneği” sayfa 112
- “SMOKIN örnekleri” sayfa 148
- “Ölü-harfli kuyruk işleyicisi örneği” sayfa 121
- “Connect örnek programı” sayfa 112
- “API çıkış örnek programı” sayfa 108
- “ Windows sistemlerinde SSPI güvenlik çıkışısının kullanılması” sayfa 162
- “Uzak kuyruklar kullanarak örnekleri çalıştırma” sayfa 163
- “Küme Kuyruğu İzleme örnek programı (AMQSCLM)” sayfa 163
- “Bağlantı Uç Noktası Araması örnek programı (CEPL)” sayfa 172

Örnek programlar içinde gösterilen özellikler

WebSphere MQ örnek programları tarafından gösterilen teknikleri gösteren tablolardan oluşan bir derlem.

MQOPER ve MQCLOSE çağrılarını kullanan tüm örnekler açık ve kapalı kuyruklar, bu nedenle bu teknikler tablolarda ayrı olarak listelenmez. İlgilendiğiniz platformu içeren başlığa bakın.

UNIX and Linux sistemleri için örnekler

Bu konuda, UNIX and Linux sistemlerinde WebSphere MQ için örnek programlar tarafından gösterilen teknikler gösterilmektedir.

UNIX ve Linux sistemlerinde WebSphere MQ için örnek programların nerede depolandığı öğrenmek için “Örnek programların UNIX sistemleri üzerinde hazırlanması ve çalıştırılması” sayfa 106 ' i (bkz..) bakın.

Çizelge 14 sayfa 94 Tablo, hangi C ve COBOL kaynak dosyalarının sağlanıp sağlanmadığını ve bir sunucunun ya da istemci yürütülebilir dosyasının dahil edilip edilip eklenmeyeceğini listeler.

Çizelge 14. WebSphere MQ on UNIX and Linux , MQI (C ve COBOL) kullanımını gösteren örnek programlar

Teknik	**C (kaynak) ("1" sayfa 96)**	**COBOL (kaynak) ("2" sayfa 96)**	**Sunucu (C yürütülür dosyası)**	**İstemci (C yürütülür dosyası) ("3" sayfa 96)**
Yayınlama/abone olma arabirimini kullanma	amqspuba amqssuba amqssbxa	örnek yok	amqspub amqssub amqssbx	örnek yok
MQPUT çağrısını kullanarak ileti koyma	amqsput0	amq0put0	amqsput	amqsputc
MQPUT1 çağrısını kullanarak tek bir ileti koyma	amqsinqa amqsecha	amqminqx amqmechx amqiinqx amqiechx	amqsinq amqsech	amqsechc
Dağıtım listesine ileti koyma ("4" sayfa 96)	amqsptl0	amq0ptl0.cbl	amqsptl	amqsptlc
İstek iletisi yanıtlama	amqsinqa	amqminqx amqiinqx	amqsinq	örnek yok
İleti alınıyor (bekleme yok)	amqsgbr0	amq0gbr0	amqsgbr	örnek yok
İletileri alma (bir süre sınırlaması ile bekleme)	amqsget0	amq0get0	amqsget	amqsgetc
İletileri alma (sınırsız bekleme)	amqstrg0	örnek yok	amqstrg	amqstrgc
İletileri alma (veri dönüştürme ile)	amqsecha	örnek yok	amqsech	örnek yok
Bir Kuyruğa Başvuru Iletileri Koyma ("4" sayfa 96)	amqsprma	örnek yok	amqsprm	amqsprmc
Kuyruktan Başvuru İletileri Alınması ("4" sayfa 96)	amqsgrma	örnek yok	amqsgrm	amqsgrmc
Başvuru İletisi kanal çıkışı ("4" sayfa 96)	amqsqrma amqsxrma	örnek yok	amqsxrm	örnek yok
İletinin ilk 20 karakterine göz atma	amqsgbr0	amq0gbr0	amqsgbr	amqsgbrc
Tüm iletilere göz atma	amqsbcg0	örnek yok	amqsbcg	amqsbcgc
Paylaşılan giriş kuyruğunun kullanılması	amqsinqa	amqminqx amqiinqx	amqsinq	amqsinqc
Dışlayıcı bir giriş kuyruğunu kullanma	amqstrg0	amq0req0	amqstrg	amqstrgc
MQINQ çağrısının kullanılması	amqsinqa	amqminqx amqiinqx	amqsinq	örnek yok
MQSET çağrısının kullanılması	amqsseta	amqmsetx amqisetx	amqsset	amqssetc
Yanıtlama Kuyruğu Kullanılması	amqsreq0	amq0req0	amqsreq	amqsreqc
İleti kural dışı durumları isteniyor	amqsreq0	amq0req0	amqsreq	örnek yok
Kesilmiş bir iletinin kabul edilmesi	amqsgbr0	amq0gbr0	amqsgbr	örnek yok
Çözülmüş bir kuyruk adının kullanılması	amqsgbr0	amq0gbr0	amqsgbr	örnek yok
Süreci tetikleme	amqstrg0	örnek yok	amqstrg	amqstrgc

Çizelge 14. WebSphere MQ on UNIX and Linux , MQI (C ve COBOL) kullanımını gösteren örnek programlar (devamı var)

Teknik	**C (kaynak) (“1” sayfa 96)**	**COBOL (kaynak) (“2” sayfa 96)**	**Sunucu (C yürütülür dosyası)**	**İstemci (C yürütülür dosyası) (“3” sayfa 96)**
Veri dönüştürmenin kullanılması	(“5” sayfa 96)	örnek yok	örnek yok	örnek yok
WebSphere MQ (XA uyumlu veritabanı yöneticilerini eşgüdümleme) SQL kullanarak tek bir veritabanına erişilmesi	amqsxas0.sqc DB2 amqsxas0.ec Informix	amq0xas0.sqb	örnek yok	örnek yok
WebSphere MQ (XA uyumlu veritabanı yöneticileri eşgüdümü) SQL kullanarak iki veritabanına erişir	amqsxag0.c amqsxab0.sqc amqsxaf0.sqc	amq0xag0.cbl amq0xab0.sqb amq0xaf0.sqb	örnek yok	örnek yok
CICS hareketi (“6” sayfa 96)	amqscic0.ccs	örnek yok	amqscic0	örnek yok
Encina işlemi (“4” sayfa 96)	amqsxae0	örnek yok	amqsxae0	örnek yok
İletileri yerleştirmek için SMOKIN işlemi (“7” sayfa 96)	amqstxpx	örnek yok	örnek yok	örnek yok
İletileri almak için SMOKIN işlemi (“7” sayfa 96)	amqstxgx	örnek yok	örnek yok	örnek yok
SMOKIN sunucusu (“7” sayfa 96)	amqstxsx	örnek yok	örnek yok	örnek yok
Kuyruk-harf kuyruğu işleyicisi	Dizin `./tools/c/Samples/dlq` (“8” sayfa 96)	örnek yok	amqsdlq	örnek yok
Bir MQI istemcisinden bir ileti yerleştirerek	örnek yok	örnek yok	örnek yok	amqsputc
Bir MQI istemcisinden ileti alma	örnek yok	örnek yok	örnek yok	amqsgetc
MQCONNX kullanılarak kuyruk yöneticiyle bağlantı kuruluyor	amqscnxc	örnek yok	örnek yok	amqscnxc
API çıkışlarının kullanılması	amqsaxe0	örnek yok	amqsaxe	örnek yok
Küme iş yükü dengeleme çıkışı	amqswlm0	örnek yok	amqswlm	örnek yok
MQSTAT çağrısını kullanarak iletileri zamanuyumsuz olarak alma ve durum alma	amqsapt0	örnek yok	amqsapt	amqsaptc
Yeniden bağlanabilir istemciler	amqsphac amqsghac amqsmhac	örnek yok	geçerli değil	amqsphac amqsghac amqsmhac
Birden çok kuyruktan iletileri zamanuyumsuz olarak tüketmek için ileti tüketicilerinin kullanılması	amqscbf0	örnek yok	amqscbf	amqscbfc
MQCONNX üzerinde SSL/TLS bağlantı bilgilerinin belirtilmesi	amqsssl c	örnek yok	geçerli değil	amqsssl c

Çizelge 14. WebSphere MQ on UNIX and Linux , MQI (C ve COBOL) kullanımını gösteren örnek programlar (devamı var)

Teknik	C (kaynak) (“1” sayfa 96)	COBOL (kaynak) (“2” sayfa 96)	Sunucu (C yürütülür dosyası)	İstemci (C yürütülür dosyası) (“3” sayfa 96)

Notlar:

1. WebSphere MQ MQI istemcisi örneklerinin yürütülebilir sürümü, bir sunucu ortamında çalışan örneklerle aynı kaynağı paylaşır.
2. Micro Focus COBOL derleyicisi ile 'amqm' başlayan programları, IBM COBOL derleyicisiyle başlayan 'amqi' ve 'amq0' başlangıclarıyla başlayan 'amqm' programını kullanın.
3. WebSphere MQ MQI istemci örneklerinin yürütülebilir sürümleri WebSphere MQ ' da HP-UXiçin kullanılabilir değildir.
4. WebSphere MQ for AIX, WebSphere MQ for HP-UX, and WebSphere MQ for Solaris üzerinde desteklenir.
5. WebSphere MQ for AIX, WebSphere MQ for HP-UX, and WebSphere MQ for Solaris bu program `amqsvfc0.`colarak adlandırılır.
6. CICS is supported by WebSphere MQ for AIX and WebSphere MQ for HP-UX only.
7. SMOKIN, System p üzerinde Linux için WebSphere MQ tarafından desteklenmiyor.
8. Ölü-harfli kuyruk işleyicisi için kaynak birkaç dosyadan oluşur ve ayrı bir dizinde sağlanır.

UNIX and Linux sistemlerine ilişkin destek hakkında ayrıntılı bilgi için IBM WebSphere MQadresindeki WebSphere MQ sistem gereksinimleri sayfasında yer alan bilgilere erişebilirsiniz.

Samples for IBM WebSphere MQ client for HP Integrity NonStop Server

Bu konuda, HP Integrity NonStop Server sistemlerinde IBM WebSphere MQ istemcisi için örnek programlar tarafından gösterilen teknikler gösterilmektedir.

Çizelge 15 sayfa 96Tabloda, sağlanan C, COBOL ve pTAL kaynak örnek programları listelenir.

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL

Teknik	C				COBOL		pTAL	
	OSS (Kaynak)	OSS (Yürütülebilir)	Guardian (Kaynak)	Guardian (Yürütülebilir)	OSS (Kaynak)	Guardian (Kaynak)	OSS (Kaynak)	Guardian (Kaynak)
Yayınlama/abone olma arabirimini kullanma	amqspuba.c amqssbxa.c amqssuba.c amqspse0.c	amqspubc amqssbxc amqssubc	MQSPUBC MQSSBXC MQSSUBC	AMQSPUBC AMQSSBXC AMQSSUBC	amq0pub0.cbl amq0sub0.cbl	MQSPUBL MQSSUBL	amqtpub0.tal amqtsub0.tal	MQSPUBT MQSSUBT
MQPUT çağrısını kullanarak ileti koyma	amqsput0.c	amqsputc	MQSPUTC	AMQSPUTC	amq0put0.cbl	MQSPUTL	amqtput0.tal	MQSPUTT

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL (devamı var)

Teknik	C				COBOL		pTAL	
MQPUT1 çağrısını kullanarak tek bir ileti koyma	amqsecha.c	amqsechc	MQSECHC	AMQSECHC			amqtech0.tal	MQSECHT
İletileri Dağıtım Listesine Koyma	amqsptl0.c	amqsptlc	MQSPTLC	AMQSPTLC	amq0ptl0.cbl	MQSPTLL		
İstek iletisi yanıtlama	amqsinqa.c	amqsinqc	MQSINQC	AMQSINQC				
İleti alınıyor (bekleme yok)	amqsgbr0.c	amqsgbrc	MQSGBRC	AMQSGBRC	amq0gbr0.cbl	MQSGBRL		
İletileri alma (bir süre sınırlaması ile bekleme)	amqsget0.c	amqsgetc	MQSGETC	AMQSGETC	amq0get0.cbl	MQSGEL	amqtget0.tal	MQSGETT
İletileri alma (sınırsız bekleme)	amqstrg0.c	amqstrgc	MQSTRGC	AMQSTRGC				
İletileri alma (veri dönüştürme ile)	amqsecha.c	amqsechc	MQSECHC	AMQSECHC				
Başvuru İletilerini Kuyruğa Koyma	amqsprma.c	amqsprmc	MQSPRMC	AMQSPRMC				
Kuyruktan Başvuru İletileri Alınması	amqsgrma.c	amqsgrmc	MQSGRMC	AMQSGRMC				
Başvuru İletisi kanal çıkışı	amqsqrma.c amqsxrma.c		MQSQRMC MQSXRMC					

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL (devamı var)

Teknik	C				COBOL		pTAL	
İletinin ilk 20 karakterine göz atma	amqsgbr0.c	amqsgbrc	MQSGBRC	AMQSGBRC	amq0gbr0.cbl	MQSGBRL		
Tüm iletilere göz atma	amqsbcg0.c	amqsbcgc	MQSBCGC	AMQSBCGC				
Paylaşılan giriş kuyruğunun kullanılması	amqsinqa.c	amqsinqc	MQSINQC	MQSINQC				
Dışlayıcı bir giriş kuyruğunu kullanma	amqstrg0.c	amqstrgc	MQSTRGC	AMQSTRGC				
MQINQ çağrısının kullanılması	amqsinqa.c	amqsinqc	MQSINQC	AMQSINQC				
MQSET çağrısının kullanılması	amqsseta.c	amqssetc	MQSSETC	AMQSSETC				
Yanıtlama Kuyruğu Kullanılması	amqsreq0.c	amqsreqc	MQSREQC	AMQSREQC	amq0req0.cbl	MQSREQL		
İleti kural dışı durumlarını isteniyor	amqsreq0.c	amqsreqc	MQSREQC	AMQSREQC	amq0req0.cbl	MQSREQL		
Kesilmiş bir iletinin kabul edilmesi	amqsgbr0.c	amqsgbrc	MQSGBRC	AMQSGBRC	amq0gbr0.cbl	MQSGBRL		

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL (devamı var)

Teknik	C				COBOL		pTAL	
Çözülmüş bir kuyruk adının kullanılması	amqsgbr0.c	amqsgbrc	MQSGBRC	AMQSGBRC	amq0gbr0.cbl	MQSGBRL		
Süreci tetiklemе	amqstrg0.c	amqstrgc	MQSTRGC	AMQSTRGC				
Veri dönüştürmenin kullanılması	amqsvfc0.c							
Ölü harf kuyruğu işleyicisi (1)	Dizin ./samp/dlq							
MQCONNX kullanılarak kuyruk yöneticiyle bağlantı kuruluyor	amqscnxc.c	amqscnxc	MQCNXC					
API çıkışlarının kullanılması	amqsaxe0.c amqsaem0.c							
Küme iş yükü dengeleme çıkışı	amqswlm0.c		MQSWLMC					
Küme kuyruğu monitörü	amqsclma.c							

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL (devamı var)

Teknik	C				COBOL		pTAL	
MQSTAT çağrısını kullanarak iletileri zamanuyumsuz olarak alma ve durum alma	amqsapt0.c	amqsaptc	MQSAPTC	MQSAPTC				
Yeniden bağlanabilir istemciler	amqsghac.c amqsmhac.c amqsphac.c	amqsghac amqsmhac amqsphac	MQSGHAC MQSMHAC MQSPHAC MQSFHAC	AMQSGHAC AMQSMHAC AMQSPHAC AMQSFHAC				
İletilerin birden çok kuyruktan iletileri zamanuyumsuz olarak tüketmesi için ileti tüketicilerinin kullanılması	amqscbf0.c	amqscbfc						
MQCONNX üzerinde SSL/TLS bağlantı bilgilerinin belirtilmesi	amqsssslc.c	amqsssslc	MQSSSLC	AMQSSSLC				
Etkinlik izleme	amqsact0.c	amqsactc	MQSACTC	AMQSACTC				
İleti Özellikleri	amqsiqma.c amqsstma.c	amqsiqmc amqsstmc	MQSIQMC MQSSTMC	AMQSIQMC AMQSSTMC				

Çizelge 15. IBM WebSphere MQ on HP Integrity NonStop Server sample programs demonstrating use of C, COBOL, and pTAL (devamı var)

Teknik	C				COBOL		pTAL	
Komut sunucusu	amqsstop.c		MQSSTOC					
Günlük Olayları	amqslog0.c	amqslogc	MQSLOGC	AMQSLOGC				
Muhasebe	amqsmon0.c	amqsmonc	MQSMONC	AMQSMONC				
Denetim arabirimi	amqsaicq.c amqsaiem.c amqsailq.c							
pTALçağrılmasına ilişkin C dili ana işlevine ilişkin bir örnek			MQSPTMC					

Notlar:

1. Ölü-harfli kuyruk işleyicisi için kaynak birkaç dosyadan oluşur ve ayrı bir dizinde sağlanır.
2. HP Integrity NonStop Server platformunda IBM WebSphere MQ istemciniz için uygulamalar geliştirmeyle ilgili bilgi edinmek için aşağıdaki başlara bakın:
   - "Uygulamanızı HP Integrity NonStop Serverüzerinde oluşturma" sayfa 416
     - "C programlarını HP Integrity NonStop Serveriçinde hazırlama" sayfa 418
     - "COBOL programlarının hazırlanması" sayfa 419
     - "pTAL programlarının hazırlanması" sayfa 420

Samples for IBM WebSphere MQ for Pencereler

This topic shows the techniques demonstrated by the sample programs for IBM WebSphere MQ for Pencereler.

Çizelge 16 sayfa 101Tablo, hangi C ve COBOL kaynak dosyalarının sağlanıp sağlanmadığını ve bir sunucunun ya da istemci yürütülebilir dosyasının dahil edilip edilip eklenmeyeceğini listeler.

Çizelge 16. MQI (C ve COBOL) kullanımını gösteren Windows örnek programları içinIBM WebSphere MQ

Teknik	C (kaynak)	COBOL (kaynak)	Sunucu (C yürütülür dosyası)	İstemci (C yürütülür dosyası)
Yayınlama/abone olma arabirimini kullanma	amqspuba amqssuba amqssbxa	örnek yok	amqspub amqssub amqssbx	örnek yok
MQPUT çağrısını kullanarak ileti koyma	amqsput0	amq0put0	amqsput	amqsputc

Çizelge 16. MQI (C ve COBOL) kullanımını gösteren Windows örnek programları içinIBM WebSphere MQ (devamı var)

Teknik	**C (kaynak)**	**COBOL (kaynak)**	**Sunucu (C yürütülür dosyası)**	**İstemci (C yürütülür dosyası)**
MQPUT1 çağrısını kullanarak tek bir ileti koyma	amqsinqa amqsecha	amqminq2 amqmech2 amqiinq2 amqiech2	amqsinq amqsech	amqsinqc amqsechc
İletileri Dağıtım Listesine Koyma	amqsptl0	amq0ptl0.cbl	amqsptl	amqsptlc
İstek iletisi yanıtlama	amqsinqa	amqminq2 amqiinq2	amqsinq	amqsinqc
İleti alınıyor (bekleme yok)	amqsgbr0	amq0gbr0	amqsgbr	amqsgbrc
İletileri alma (bir süre sınırlaması ile bekleme)	amqsget0	amq0get0	amqsget	amqsgetc
İletileri alma (sınırsız bekleme)	amqstrg0	örnek yok	amqstrg	amqstrgc
İletileri alma (veri dönüştürme ile)	amqsecha	örnek yok	amqsech	amqsechc
Başvuru İletilerini Kuyruğa Koyma	amqsprma	örnek yok	amqsprm	amqsprmc
Kuyruktan Başvuru İletileri Alınması	amqsgrma	örnek yok	amqsgrm	amqsgrmc
Başvuru İletisi kanal çıkışı	amqsqrma amqsxrma	örnek yok	amqsxrm	örnek yok
İletinin ilk 20 karakterine göz atma	amqsgbr0	amq0gbr0	amqsgbr	amqsgbrc
Tüm iletilere göz atma	amqsbcg0	örnek yok	amqsbcg	amqsbcgc
Paylaşılan giriş kuyruğunun kullanılması	amqsinqa	amqminq2 amqiinq2	amqsinq	amqsinqc
Dışlayıcı bir giriş kuyruğunu kullanma	amqstrg0	amq0req0	amqstrg	amqstrgc
MQINQ çağrısının kullanılması	amqsinqa	amqminq2 amqiinq2	amqsinq	amqsinqc
MQSET çağrısının kullanılması	amqsseta	amqmset2 amqiset2	amqsset	amqssetc
MQINQMP çağrısının kullanılması	amqsiqma	örnek yok	örnek yok	örnek yok
Yanıtlama Kuyruğu Kullanılması	amqsreq0	amq0req0	amqsreq	amqsreqc
İleti kural dışı durumları isteniyor	amqsreq0	amq0req0	amqsreq	amqsreqc
Kesilmiş bir iletinin kabul edilmesi	amqsgbr0	amq0gbr0	amqsgbr	amqsgbrc
Çözülmüş bir kuyruk adının kullanılması	amqsgbr0	amq0gbr0	amqsgbr	amqsgbrc
Süreci tetikleme	amqstrg0	örnek yok	amqstrg	amqstrgc
Veri dönüştürmenin kullanılması	amqsvfc0	örnek yok	örnek yok	örnek yok
WebSphere MQ (XA uyumlu veritabanı yöneticilerini eşgüdümleme) SQL kullanarak tek bir veritabanına erişilmesi	amqsxas0.sqc DB2 amqsxas0.ec Informix	amq0xas0.sqb	örnek yok	örnek yok

Çizelge 16. MQI (C ve COBOL) kullanımını gösteren Windows örnek programları içinIBM WebSphere MQ (devamı var)

Teknik	**C (kaynak)**	**COBOL (kaynak)**	**Sunucu (C yürütülür dosyası)**	**İstemci (C yürütülür dosyası)**
WebSphere MQ (XA uyumlu veritabanı yöneticileri eşgüdümü) SQL kullanarak iki veritabanına erişir	amqsxag0.c amqsxab0.sqc DB2 amqsxaf0.sqc DB2	amq0xag0.cbl amq0xab0.sqb amq0xaf0.sqb	örnek yok	örnek yok
İletileri koymak için SMOKIN işlemi	amqstxpx	örnek yok	örnek yok	örnek yok
İletileri almak için SMOKIN işlemi	amqstxgx	örnek yok	örnek yok	örnek yok
SMOKIN sunucusu	amqstxsx	örnek yok	örnek yok	örnek yok
Kuyruk-harf kuyruğu işleyicisi	Dizin `./tools/c/Samples/dlq` (“1” sayfa 103)	örnek yok	amqsdlq	örnek yok
Bir WebSphere MQ MQI istemcisinden bir ileti yerleştirerek	örnek yok	örnek yok	örnek yok	amqsputc
Bir WebSphere MQ MQI istemcisinden bir ileti alma	örnek yok	örnek yok	örnek yok	amqsgetc
MQCONNX kullanılarak kuyruk yöneticiyle bağlantı kuruluyor	amqscnxc	örnek yok	örnek yok	amqscnxc
API çıkışlarının kullanılması	amqsaxe0	örnek yok	amqsaxe	örnek yok
Küme iş yükü dengeleme	amqswlm0	örnek yok	amqswlm	örnek yok
SSPI güvenlik yordamları	amqsspin	örnek yok	amqrspin.dll	amqrspin.dll
MQSTAT çağrısını kullanarak iletileri zamanuyumsuz olarak alma ve durum alma	amqsapt0	örnek yok	amqsapt	amqsaptc
Yeniden bağlanabilir istemciler	amqsphac amqsghac amqsmhac	örnek yok	Burada geçerli değil	amqsphac amqsghac amqsmhac
Birden çok kuyruktan iletileri zamanuyumsuz olarak tüketmek için ileti tüketicilerinin kullanılması	amqscbf0	örnek yok	amqscbf	amqscbfc
MQCONNX üzerinde SSL/TLS bağlantı bilgilerinin belirtilmesi	amqssslc	örnek yok	geçerli değil	amqssslc

Notlar:

1. Ölü-harfli kuyruk işleyicisi için kaynak birkaç dosyadan oluşur ve ayrı bir dizinde sağlanır.

Visual Basic Samples for IBM WebSphere MQ for Windows

This topic shows the techniques demonstrated by the Visual Basic sample programs for IBM WebSphere MQ for Pencereler.

Çizelge 17 sayfa 104 shows the techniques demonstrated by the IBM WebSphere MQ for Pencereler sample programs.

Bir proje birkaç dosya içerebilir. Visual Basic i‡inde bir proje açmanken, di§er ktklerin otomatik olarak yklenir. Yürütülebilir bir program programı sağlanmaz.

mqtrivc.vbpdışında tüm örnek projeler IBM WebSphere MQ Server ile çalışacak şekilde ayarlanır. To find out how to change the sample projects to work with the IBM WebSphere MQ clients see “Visual Basic Programlarinin Windows 'ta Hazırlanması” sayfa 443.

Çizelge 17. IBM WebSphere MQ for Windows için MQI (Visual Basic) kullanımı gösteren örnek programlar

Teknik	**Proje dosyası adı**
MQPUT çağrısını kullanarak ileti koyma	amqsputb.vbp
MQGET çağrısını kullanarak ileti alınması	amqsgetb.vbp
MQGET çağrısını kullanarak kuyruğa göz atma	amqsbcgb.vbp
Basit MQGET ve MQPUT örneği (istemci)	mqtrivc.vbp
Basit MQGET ve MQPUT örneği (sunucu)	mqtrivs.vbp
Dizgileri ve kullanıcı tanımlı yapıları MQPUT ve MQGET kullanarak koyma ve alma	strings.vbp
Bir kanalı başlatmak ve durdurmak için PCF yapılarının kullanılması	pcfsamp.vbp
MQAI kullanılarak kuyruk yaratılması	amqsaicq.vbp
MQAI kullanarak kuyruk yöneticisinin kuyrukları listeleniyor	amqsailq.vbp
MQAI kullanarak olayları izleme	amqsaiem.vbp

Örnek programların hazırlanması ve çalıştırılması

Kuyruk yöneticinizi, istemci kipinde çalışan uygulamalardan gelen bağlantı isteklerini güvenli bir şekilde kabul etmek üzere yapılandırın.

Başlamadan önce

Kuyruk yöneticisinin zaten var olduğundan ve başlatıldığından emin olun. Kanal doğrulama kayıtlarının önceden aşağıdaki gibi etkinleştirilip etkinleştirilmediğini belirleyin:

```
DISPLAY QMGR CHLAUTH
```

Bu görev, kanal doğrulama kayıtlarının etkinleştirilmesini bekler. Bu, diğer kullanıcılar ve uygulamalar tarafından kullanılan bir kuyruk yöneticisiyse, bu ayarın değiştirilmesi diğer tüm kullanıcıları ve uygulamaları etkiler. If your queue manager does not make use of channel authentication records then step “4” sayfa 105 can be replaced with an alternate authentication method (for example a security exit) which sets the MCAUSER to the *ayrıcalıksız-kullanıcı kimliği* you will obtain in step “1” sayfa 105.

Uygulamanızın hangi kanal adını kullanmayı beklediğini bilmeniz gerekir. Böylece, uygulamanın kanalı kullanmasına izin verilmelidir. Ayrıca, hangi nesnelerin, örneğin kuyruklar ya da konular için uygulamanızın kullanılmasını beklediğinden, uygulamanızın bunları kullanmasına izin verilebilmesi için izin verileceğini de bilmeniz gerekir.

Bu görev hakkında

Bu görev, kuyruk yöneticisine bağlanan bir istemci uygulaması için kullanılmak üzere ayrıcalıklı olmayan bir kullanıcı kimliği yaratır. İstemci uygulaması için erişim izni verilir; yalnızca, gereken kanalı ve bu kullanıcı kimliğini kullanarak gereksinim duyduğu kuyruğu kullanabilmelidir.

Yordam

1. Kuyruk yöneticinizin çalışmakta olduğu sistemde bir kullanıcı kimliği edinin. Bu görev için bu kullanıcı kimliği ayrıcalıklı bir yönetimle görevli kullanıcı olmamalıdır. Bu kullanıcı kimliği, istemci bağlantısının kuyruk yöneticisi üzerinde çalışacağı yetki olacaktır.
2. Aşağıdaki komutlarla bir dinleyici programı başlatın:

   qmgr kuyruk yöneticinizin adıdır
   nnnn , seçtiğiniz kapı numarasıdır

   a) UNIX ve Windows sistemleri için:

   ```
   runmqlsr -t tcp -m qmgr -p nnnn
   ```

3. Uygulamanınız SYSTEM.DEF.SVRCONN daha sonra bu kanal önceden tanımlıdır. Uygulamanız başka bir kanal kullanıyorsa, MQSC komutunu kullanarak bu kanalı yaratın:

   ```
   DEFINE CHANNEL('channel-name') CHLTYPE(SVRCONN) TRPTYPE(TCP) +
   DESCR('Channel for use by sample programs')
   ```

   kanal-adı , kanalınızın adıdır.
4. İstemci sisteminizin yalnızca IP adresinin, bu kanalı kullanarak, MQSC komutunu vererek kanal kullanmasını sağlayacak bir kanal doğrulama kuralı yaratın:

   ```
   SET CHLAUTH('channel-name') TYPE(ADDRESSMAP) ADDRESS('client-machine-IP-address') +
   MCAUSER('non-privileged-user-id')
   ```

   kanal-adı , kanalınızın adıdır.
   client-machine-IP-address , istemci sisteminizin IP adresidir.
   Örnek istemci uygulamanız kuyruk yöneticiyle aynı makinede çalıştırılıyorsa, uygulamanız 'localhost' ile bağlantı kuracaksa '127.0.0.1' IP adresini kullanın. Birden çok farklı istemci makinesi bağlanacaksa, tek bir IP adresi yerine bir kalıp ya da aralık kullanabilirsiniz. Ayrıntılı bilgi için Soysal IP adresleri başlıklı konuya bakın.
   ayrıcalıklı olmayan-kullanıcı-kimliği , “1” sayfa 105adımında edindiğiniz kullanıcı kimliğidir.
5. Uygulamanınız SYSTEM.DEFAULT.LOCAL.QUEUE (Kuyruk), bu kuyruk zaten tanımlı. Uygulamanınız başka bir kuyruk kullanıyorsa, MQSC komutunu vererek yaratın:

   ```
   DEFINE QLOCAL('queue-name') DESCR('Queue for use by sample programs')
   ```

   kuyruk-adı , kuyruğunuzun adıdır.
6. Kuyruk yöneticisine bağlanmak ve sorgulamak için erişim izni verin:

   a) UNIX veWindows sistemleri için, MQSC komutları yayınıdır:

   ```
   SET AUTHREC OBJTYPE(QMGR) PRINCIPAL('non-privileged-user-id') +
   AUTHADD(CONNECT, INQ)
   ```

   ayrıcalıklı olmayan-kullanıcı-kimliği , “1” sayfa 105adımında edindiğiniz kullanıcı kimliğidir.
7. Uygulamanız noktadan noktaya iletişim uygulamasıysa, bu, kuyrukların kullanılmasını sağlar ve MQSC komutları vererek, kullanılacak kullanıcı kimliği tarafından kuyruğunuzu kullanarak kuyruğa alma ve iletileri alma ve iletileri alma izni verir.

   a) UNIX ve Windows sistemleri için, MQSC komutları yayınıdır:

   ```
   SET AUTHREC PROFILE('queue-name') OBJTYPE(QUEUE) +
   PRINCIPAL('non-privileged-user-id') AUTHADD(PUT, GET, INQ, BROWSE)
   ```

   kuyruk-adı , kuyruğunuzun adıdır.
   ayrıcalıklı olmayan-kullanıcı-kimliği , “1” sayfa 105adımında edindiğiniz kullanıcı kimliğidir.
8. Uygulamanız bir yayınlama/abone olma uygulamasıysa, bu, konuların kullanılmasını sağlar, MQSC komutları vererek, kullanılacak kullanıcı kimliği ile konularınızı kullanarak yayınlamaya ve abone olmaya izin verir.

a) UNIX ve Windows sistemleri için, MQSC komutları yayınıdır:

```
SET AUTHREC PROFILE('SYSTEM.BASE.TOPIC') OBJTYPE(TOPIC) +
PRINCIPAL('non-privileged-user-id') AUTHADD(PUB, SUB)
```

ayrıcalıklı olmayan-kullanıcı-kimliği , “1” sayfa 105adımında edindiğiniz kullanıcı kimliğidir. Bu işlem, konu ağacındaki herhangi bir konuya *ayrıcalıksız-kullanıcı-kimliği* erişimi verir; diğer bir seçenek olarak, **DEFINE TOPIC** kullanarak bir konu nesnesi tanımlayabilir ve yalnızca o konu nesnesinin gönderme yaptığı konu ağacının bir bölümüne erişimler verir. Ayrıntılı bilgi için Konulara kullanıcı erişiminin denetlenmesi başlıklı konuya bakın.

Sonraki adım

Artık istemci uygulamanız kuyruk yöneticisine bağlanabilir ve kuyruğu kullanarak ileti alabilir ya da alabilir.

İlgili görevler

UNIX ya da Linux sistemlerinde ve Windows sistemlerinde WebSphere MQ nesnesine erişim verilmesi

İlgili başvurular

CHLAUTH KüMESI

KANAL TANIMLA

QLOCAL ' I TANIMLA

AUTHREC

Örnek programların UNIX sistemleri üzerinde hazırlanması ve çalıştırılması

Çizelge 18. UNIX and Linux sistemlerinde WebSphere MQ için örneklerin nerede bulunması gerekir

İçerik	**Dizin**
Kaynak dosyalar	*MQ_INSTALLATION_PATH*/samp
Ölü-harfli kuyruk işleyicisi kaynak dosyaları	*MQ_INSTALLATION_PATH*/samp/dlq
yürütülür dosyalar	*MQ_INSTALLATION_PATH*/samp/bin
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

The WebSphere MQ on UNIX and Linux systems sample files are in the directories listed in Çizelge 18 sayfa 106 if the defaults were used at installation time. Örnekleri çalıştırmak için, sağlanan yürütülebilir sürümleri kullanın ya da bir ANSI derleyicisini kullanarak, kaynak sürümlerini başka uygulamalar gibi derleyin. Bunun nasıl yapacağına ilişkin bilgi için bkz. “Örnek programların çalıştırılması” sayfa 107.

Örnek programların Windows sistemleri üzerinde hazırlanması ve çalıştırılması

Çizelge 19. Where to find the samples for WebSphere MQ for Pencereler

İçerik	**Dizin**
C kaynak kodu	*MQ_INSTALLATION_PATH*\Tools\C\Örnekler
Ölü-mektup işleyici örneği için kaynak kod	*MQ_INSTALLATION_PATH*\Tools\C\Samples\DLQ
COBOL kaynak kodu	*MQ_INSTALLATION_PATH*\Tools\Cobol \ Örnekler
C yürütülür dosyaları [1]	*MQ_INSTALLATION_PATH*\ Tools\C\Samples \ Bin (32 bit sürümler) *MQ_INSTALLATION_PATH*\ Tools\C\Samples\Bin64 (64-bit sürümler)
Örnek MQSC dosyaları	*MQ_INSTALLATION_PATH*\Tools\MQSC\Samples
Visual Basic kaynak kodu	*MQ_INSTALLATION_PATH*\Tools\VB\SampVB6
.NET örnekleri	*MQ_INSTALLATION_PATH*\Tools\dotnet \ Örnekler

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Not:

1. 64 bit sürümler bazı C yürütülebilir dosya örneklerinden kullanılabilir.

The WebSphere MQ for Pencereler sample files are in the directories listed in Çizelge 19 sayfa 106 if the defaults were used at installation time; the installation drive defaults to <c:>. To run the samples, either use the executable versions supplied or compile the source versions as you would any other WebSphere MQ for Pencereler applications. Bunun nasıl yapacagına ilişkin bilgi için bkz. “Örnek programların çalıştırılması” sayfa 107.

Örnek programların çalıştırılması

Farklı platformlarda örnek programları çalıştırırken bu konuyu kullanmayı düşünün.

Örnek programlardan herhangi birini çalıştırabilmeniz için, bir kuyruk yöneticisi yaratın ve varsayılan tanımlamaları ayarlayın. Bu, Yönetmeiçinde açıklanır.

Windows, UNIX ve Linux platformlarında

Örneklerin çalışması için bir kuyruk kümesi gerekir. Kendi kuyruklarınızı kullanın ya da bir küme yaratmak için örnek MQSC dosyasını amqscos0.tst çalıştırın.

Bunu UNIX and Linux sistemlerinde yapmak için şunu girin:

- `runmqsc QManagerName <amqscos0.tst >/tmp/sampobj.out`

Hata olmadığından emin olmak için `sampobj.out` dosyasını denetleyin.

Bu işlemi Windows sistemlerinde yapmak için şunu girin:

- `runmqsc QManagerName  <amqscos0.tst > sampobj.out`

Hata olmadığından emin olmak için `sampobj.out` dosyasını denetleyin. Bu dosya geçerli dizininizde.

Şimdi örnek uygulamaları çalıştırabilirsiniz. Örnek uygulamanın adını ve onu izleyen herhangi bir parametrenin adını girin; örneğin:

- `amqsput myqueue qmanagername`

Burada `myqueue` , iletilerin yerleştirilecek kuyruğun adıdır; `qmanagername` ise, `myqueue`' un sahibi olan kuyruk yöneticidir.

Her bir kişinin beklediği parametrelerle ilgili bilgi için tek tek örneklerin açıklamalarına bakın.

Kuyruk adı uzunluğu

COBOL örnek programları için, kuyruk adlarını parametre olarak geçirdiğinizde, 48 karakter sağlamanız gerekir, gerekirse boş karakterlere sahip olarak doldurma yapmanız gerekir. 48 karakterden başka bir değer, programın 2085. neden koduyla başarısız olmasına neden olur.

Sorgula, Set, Echo örnekleri

Sorgula, Set, and Echo örnekleri için, örnek tanımlamaları bu örneklerin C sürümlerini tetikler.

COBOL sürümlerinin olmasını istiyorsanız, süreç tanımlamalarını değiştirmelisiniz:

- SYSTEM.SAMPLE.INQPROCESS
- SYSTEM.SAMPLE.SETPROCESS
- SYSTEM.SAMPLE.ECHOPROCESS

Windowssistemlerinde, UNIX and Linux sistemlerinde bunu `amqscos0.tst` kütüğünü düzenleyerek ve C yürütülür kütüğü adlarını daha önce gösterildiği gibi, **runmqsc** komutunu kullanmadan önce COBOL yürütülür kütüğü adlarına çevirerek gerçekleştirin.

API çıkış örnek programı

Örnek API çıkışı, kullanıcı tarafından belirtilen bir dosyaya MQAPI_TRACE_LOGFILE ortam değişkeninde tanımlı bir önek içeren bir MQI izleme oluşturur.

API çıkışlarına ilişkin daha fazla bilgi için bkz. “API çıkışlarının yazılması ve derlenmesi” sayfa 370.

Kaynak
amqsaxe0.c

İkili
amqsaxe

Örnek çıkışa ilişkin yapılandırma

1. Aşağıdaki bilgileri qm.ini dosyasına ekleyin.

   Windowsdışındaki platformlar

   ```
   ApiExitLocal:
      Sequence=100
      Function=EntryPoint
      Module=MQ_INSTALLATION_PATH/samp/bin/amqsaxe
      Name=SampleApiExit
   ```

   Burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQ ' in kurulu olduğu dizini temsil eder.

   Windows

   ```
   ApiExitLocal:
      Sequence=100
      Function=EntryPoint
      Module=MQ_INSTALLATION_PATH\Tools\c\Samples\bin\amqsaxe
      Name=SampleApiExit
   ```

   Burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQ ' in kurulu olduğu dizini temsil eder.

2. Ortam değişkenini ayarla

   ```
   MQAPI_TRACE_LOGFILE=/tmp/MqiTrace
   ```

3. Uygulamanızı çalıştırın.

   Çıkış dosyaları,namesgibi adlarla /tmp dizininde yaratılır: `MqiTrace.<pid>.<tid>.log`

Zamanuyumsuz tüketim örnek programı

amqscbf örnek programı, iletileri zamanuyumsuz olarak birden çok kuyruktan tüketebilmek için MQCB ve MQCTL ' nin kullanılmasını gösterir.

amqscbf, C kaynak kodu olarak sağlanır ve Windows, UNIX and Linux platformlarında ikili istemci ve sunucu yürütülebilir dosyası olarak sağlanır.

Program komut satırından başlatılır ve aşağıdaki isteğe bağlı parametreleri alır:

```
Usage: [Options] <Queue Name> { <Queue Name> }
  where Options are:
    -m <Queue Manager Name>
    -o <Open options>
    -r <Reconnect Type>
       d Reconnect Disabled
       r Reconnect
       m Reconnect Queue Manager
```

Birden çok kuyruktan ileti okumak için birden çok kuyruk adı belirtin (örnek tarafından en çok on kuyruk desteklenir.)

Not: *Yeniden bağlan tipi* yalnızca istemci programları için geçerlidir.

Örnek

The example shows amqscbf run as a server program reading one message from QL1 and then being stopped.

QL1' ta bir sınama iletisi yerleştirmek için WebSphere MQ Explorer olanağını kullanın. Enter tuşuna basarak programı durdurun.

```
C:\>amqscbf QL1
Sample AMQSCBF0 start

Press enter to end
Message Call (9 Bytes) :
Message 1

Sample AMQSCBF0 end
```

amqscbf 'in gösterdiği

Bu örnek, geliş sırasına göre birden çok kuyruktan iletilerin nasıl okunacağını gösterir. Bu, zamanuyumlu MQGET kullanarak çok daha fazla kod gerektirecektir. Zamanuyumsuz tüketim durumunda, yoklama gerekmez, iş parçacığı ve depolama yönetimi WebSphere MQtarafından gerçekleştirilir. "Gerçek bir dünya" örneğinin hatalarla başa çıkması gerekir; örnek hatalarda konsola yazılıyor.

Örnek kodda aşağıdaki adımlar bulunur:

1. Tek ileti tüketimi geri bildirme işlevini tanımlayın,

   ```
   void MessageConsumer(MQHCONN      hConn,
                        MQMD    * pMsgDesc,
                        MQGMO   * pGetMsgOpts,
                        MQBYTE  * Buffer,
                        MQCBC   * pContext)
   { ... }
   ```

2. Kuyruk yöneticisine bağlan,

   ```
   MQCONNX(QMName,&cno,&Hcon,&CompCode,&CReason);
   ```

3. Giriş kuyruklarını açın ve her birini `MessageConsumer` geri bildirme işleviyle ilişkilendirin,

   ```
   MQOPEN(Hcon,&od,O_options,&Hobj,&OpenCode,&Reason);
   cbd.CallbackFunction = MessageConsumer;
   MQCB(Hcon,MQOP_REGISTER,&cbd,Hobj,&md,&gmo,&CompCode,&Reason);
   ```

   `cbd.CallbackFunction` , her kuyruk için ayarlanması gerekmez; bu yalnızca giriş alanıdır. Ancak, farklı bir geri bildirme işlevini her kuyrukla ilişkilendirebilrsiniz.

4. İletilerin tüketimine başla,

   ```
   MQCTL(Hcon,MQOP_START,&ctlo,&CompCode,&Reason);
   ```

5. Kullanıcının Enter tuşuna bastıktan sonra ileti tüketimini durduruncaya kadar bekleyin.

   ```
   MQCTL(Hcon,MQOP_STOP,&ctlo,&CompCode,&Reason);
   ```

6. Son olarak kuyruk yöneticisinden kopuyor,

   ```
   MQDISC(&Hcon,&CompCode,&Reason);
   ```

Zamanuyumsuz Koy örnek programı

Amqsapt örneği ve Asynchronous Sput örnek programının tasarımını çalıştırma hakkında bilgi edinin.

Zamanuyumsuz koyma örnek programı, zamanuyumsuz MQPUT çağrısını kullanarak bir kuyruğa ileti koyar ve MQSTAT çağrısını kullanarak durum bilgilerini alır. Farklı platformlarda bu programın adı için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

amqsapt örneğinin çalıştırılması

Bu program en çok 6 parametre alır:

1. Hedef kuyruğun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)
3. Açma seçenekleri (isteğe bağlı)
4. Seçenekleri kapat (isteğe bağlı)
5. Hedef kuyruk yöneticisinin adı (isteğe bağlı)
6. Dinamik kuyruğun adı (isteğe bağlı)

Kuyruk yöneticisi belirtilmediyse, amqspt varsayılan kuyruk yöneticisine bağlanır.

Zamanuyumsuz Put örnek programının tasarımı

Program, iletileri koymak için hedef kuyruğu açmak için, sağlanan çıkış seçenekleri ile ya da MQOO_OUTPUT ve MQOO_FAIL_IF_QUIESCING seçenekleriyle MQOPER çağrısını kullanır.

Kuyruk açılamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler. Programı basit tutmak için, bu konuda ve sonraki MQI çağrılarında, program seçeneklerin çoğu için varsayılan değerleri kullanır.

Her giriş satırı için, program metni bir arabelleğe okur ve MQPMO_ASYNYC_response ile MQPUT çağrısını kullanır ve bu satırın metnini içeren bir veri paketi iletisi yaratılır ve zamanuyumsuz olarak hedef kuyruğa konmasını sağlar. Program, girişin sonuna ulaşıncaya kadar ya da MQPUT çağrısının başarısız olması için devam eder. Program girişin sonuna ulaşırsa, MQCLOSE çağrısını kullanarak kuyruğu kapatır.

Daha sonra program MQSTAT çağrısını yayınlar, bir MQSTS yapısını döndürür ve ileti sayısını başarıyla içeren iletileri, uyarı içeren ileti sayısını ve hata sayısını görüntüler.

Göz At örnek programları

Göz At örnek programları, MQGET çağrısını kullanarak kuyruklardaki iletilere göz atar.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Göz At örnek programının tasarımı

Program, MQOO_BROWSE seçeneğiyle MQOPEN çağrısını kullanarak hedef kuyruğu açar. Kuyruk açılamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler.

Kuyrukta yer alan her ileti için, program iletiyi kuyruktan kopyalamak için MQGET çağrısını kullanır, ardından iletide yer alan verileri görüntüler. MQGET çağrısı aşağıdaki seçenekleri kullanır:

MQGMO_BROWSE_NEXT
MQOPEN çağrısının ardından, göz atma imleci kuyrukta ilk iletiden önce mantıksal olarak konumlandırılır, bu nedenle bu seçenek, arama ilk kez yapıldığında ***ilk*** iletisinin döndürülmesine neden olur.

MQGMO_NO_BEKLEME
Kuyruğun üzerinde ileti yoksa program beklemez.

MQGMO_ACCEPT_TRUNCATED_MSG
MQGET çağrısı, sabit büyüklerin arabelleğinden birini belirtir. Bu arabellekten daha uzun bir ileti varsa, program kısaltılmış iletiyi görüntüler; bu ileti, iletinin kesildiğini bildiren bir uyarıyla birlikte görüntülenir.

Bu program, bu alanları, aldığı iletide yer alan değerlere ayarlaması nedeniyle, her MQGET çağrısından sonra MQMD yapısının *MsgId* ve *CorrelId* alanlarını nasıl temizlemeniz gerektiğini gösterir. Bu alanların temizlenmesi, art arda gelen MQGET çağrılarının iletilerin kuyrukta tutulmakta olduğu sırayla alma çağrıları anlamına gelir.

Program kuyruğun sonuna kadar devam eder; MQGET çağrısı, MQRC_NO_MSG_AVAILEABLE neden kodunu döndürür ve program bir uyarı iletisi görüntüler. MQGET çağrısının başarısız olması durumunda, program neden kodunu içeren bir hata iletisi görüntüler.

Daha sonra, program MQCLOSE çağrısını kullanarak kuyruğu kapatır.

UNIX, Linux ve Windows sistemleri

Consider using this topic when learning about Browse sample programs on UNIX, Linux and Pencereler systems.

Programın C sürümü 2 parametre alır

1. Kaynak kuyruğun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)

Kuyruk yöneticisi belirtilmediyse, varsayılan değer olarak varsayılan değer olarak bağlanır. Örneğin, aşağıdakilerden birini girin:

- `amqsgbr myqueue qmanagername`
- `amqsgbrc myqueue qmanagername`
- `amq0gbr0 myqueue`

Burada `myqueue` , iletilerin görüntüleneceği kuyruğun adıdır; `qmanagername` ise, `myqueue`' un sahibi olan kuyruk yöneticidir.

If you omit the `qmanagername`, when running the C sample, it assumes that the default queue manager owns the queue.

COBOL sürümünün herhangi bir parametresi yok. Bu, varsayılan kuyruk yöneticisine bağlanır ve çalıştırdığınız zaman sizden sorulur:

```
Please enter the name of the target queue
```

Bu durum söz sahibi olduğunda, her iletinin yalnızca ilk 50 karakteri görüntülenir ve `- - - truncated` tarafından izlenilir.

Tarayıcı örnek programı

Tarayıcı örnek programı, bir kuyruktaki tüm iletilerin ileti tanımlayıcısını ve ileti içerik alanlarını okur ve yazar.

Örnek program, sadece bir teknik göstermek için değil, bir yardımcı program olarak yazılmıştır. Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Bu program şu değiştirgeleri alır:

1. Kaynak kuyruğun adı
2. Kuyruk yöneticisinin adı
3. Özellikler için isteğe bağlı bir parametre.

Bu programa ilişkin ilk iki giriş parametresi zorunludur. Örneğin, programı aşağıdaki yöntemlerden birini kullanarak başlatın:

- `amqsbcg myqueue qmanagername`
- `amqsbcgc myqueue qmanagername`

Burada `myqueue` , iletilerin göz atılacağı kuyruğun adıdır; `qmanagername` ise, `myqueue`' un sahibi olan kuyruk yöneticidir.

Kuyruktan her bir iletiyi okur ve stdout 'a şunları yazar:

- Biçimlendirilmiş ileti tanımlayıcı alanları
- İleti verileri (onaltılı biçimde dökümü ve olas olas, karakter biţimi)

Özellik parametresine ilişkin izin verilen değerler şunlardır:

Değer	Davranış
0	Varsayılan davranış, V6için olduğu gibi. Uygulamaya teslim edilen özellikler, iletinin alındığı *PropertyControl* kuyruk özniteliğine bağlıdır.

1	Bir ileti tanıtıcısı yaratılır ve MQGET ile kullanılır. İleti tanımlayıcısında (ya da uzantıda) yer alan durumlar dışında, iletinin özellikleri ileti tanımlayıcısına benzer bir şekilde görüntülenir. Örneğin:
	`****Message properties****`
	`<property name> : <property value>`
	Ya da kullanılabilir özellik yoksa:
	`****Message properties****`
	`None`
	Sayısal değerler `printf`kullanılarak görüntülenir, dizgi değerleri tek tırnak işaretleriyle çevrilir ve bayt dizgileri, ileti tanımlayıcısına göre X ve tek tırnak işaretleriyle çevrelenir.

2	MQGMO_NO_XX_ENCODE_CASE_ONE properties belirtildi, bu nedenle yalnızca ileti tanımlayıcısı özellikleri döndürülecek.
3	İleti verilerinde tüm özelliklerin döndürülmesi içinMQGMO_PROPERTIES_FORCE_MQRFH2 belirtildi.
4	MQGMO_PROPERTIES_COMPATIBILITY belirtildi; böylece, bir sürüm 6 özelliğinin dahil edilip edilmediğine bağlı olarak tüm özellikler döndürülebilecek, tersi durumda özellikler atılır.

Program, iletinin ilk 65535 karakterini yazdırmayla sınırlandırılır ve daha uzun bir ileti okunduysa, *kısaltılmış msg* ' nin nedeni ile başarısız olur.

Bu yardımcı programdaki çıkışa bir örnek için Yönetme ' e bakın.

CICS hareket örneği

executablekaynak kodu için amqscic0.ccs adlı bir örnek CICS hareket programı ve yürütülebilir sürüm için amqscic0 adı verilir. Standart CICS olanaklarını kullanarak işlem yapabilirsiniz.

Altyapınız için gerekli olan komutlara ilişkin ayrıntılar için “IBM WebSphere MQ uygulaması oluşturulması” sayfa 409 ' e bakın.

Bu hareket, varsayılan kuyruk yöneticisinde SYSTEM.SAMPLE.CICS.WORKQUEUE iletim kuyruğundan iletileri okur ve bunları, iletinin iletim üstbilgisinde yer alan adı yerel kuyruğa yerleştirir. SYSTEM.SAMPLE.CICS.DLQ.

Not: Bu kuyrukları ve örnek giriş kuyruklarını yaratmak için örnek bir MQSC komut dosyası amqscic0.tst kullanabilirsiniz.

Connect örnek programı

Connect örnek programı, bir istemcideki MQCONNX çağrısını ve seçeneklerini keşfetmenizi sağlar. Bu örnek, MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanır, MQINQ çağrısını kullanarak kuyruk yöneticisinin adını sorgular ve görüntüler. Ayrıca, amqscnxc örneğinin çalıştırılmasıyla ilgili bilgi edinin.

Not: Connect örnek programı bir istemci örneğidir. Bir sunucuda derleyebilir ve çalıştırabilirsiniz, ancak işlev yalnızca bir istemcide anlamlıdır ve yalnızca istemci yürütülebilir dosyaları sağlanır.

amqscnxc örneğinin çalıştırılması

Connect örnek programının komut satırı sözdizimi şöyledir:

```
amqscnxc [-x ConnName [-c SvrconnChannelName]] [QMgrName]
```

Parametreler isteğe bağlıdır ve sırası QMgrNamedışında önemli değildir; bu değer, belirtilirse, son olarak gelmelidir. Değiştirgeler şunlardır:

ConnName
Sunucu kuyruk yöneticisinin TCP/IP bağlantı adı

SvrconnChannelAd
Sunucu bağlantı kanalının adı

QMgrName
Hedef kuyruk yöneticisinin adı

TCP/IP bağlantı adını belirtmezseniz, MQCONNX *ClientConnPtr* ile birlikte NULL (boş değer) olarak ayarlanır. TCP/IP bağlantı adını belirtirseniz, ancak sunucu bağlantı kanalına (ters çevirme işlemine izin verilmez) belirtirseniz, örnek SYSTEM.DEF.SVRCONN. Hedef kuyruk yöneticisini belirtmezseniz, örnek olarak belirtilen TCP/IP bağlantı adına hangi kuyruk yöneticisine bağlanıyorsa, bu örnek bağlantı kurar.

Not: Tek parametre olarak bir soru işareti girerseniz ya da yanlış parametreler girdiğinizde, programın nasıl kullanılacağını açıklayan bir ileti elde edin.

Örneği komut satırı seçenekleri olmadan çalıştırırsanız, bağlantı bilgilerini belirlemek için MQSERVER ortam değişkeninin içeriği kullanılır. (Bu örnekte MQSERVER, `SYSTEM.DEF.SVRCONN/TCP/machine.site.company.com`olarak ayarlıdır.) Şu şekilde çıktıyı görüyorsunuz:

```
Sample AMQSCNXC start
Connecting to the default queue manager
with no client connection information specified.
Connection established to queue manager machine

Sample AMQSCNXC end
```

Örneği çalıştırıp bir TCP/IP bağlantı adı ve bir sunucu bağlantısı kanal adı sağlıyorsa, ancak hedef kuyruk yöneticisi adı girmiyorsa, aşağıdaki gibi:

```
amqscnxc -x machine.site.company.com -c SYSTEM.ADMIN.SVRCONN
```

Varsayılan kuyruk yöneticisi adı kullanılır ve şu şekilde çıktıyı görürsünüz:

```
Sample AMQSCNXC start
Connecting to the default queue manager
using the server connection channel SYSTEM.ADMIN.SVRCONN
on connection name machine.site.company.com.
Connection established to queue manager MACHINE

Sample AMQSCNXC end
```

Örneği çalıştırırsanız ve bir TCP/IP bağlantı adı ve hedef kuyruk yöneticisi adı sağlıyorsa, aşağıdaki gibi:

```
amqscnxc -x machine.site.company.com MACHINE
```

Çıktıyı şu şekilde görüyorsunuz:

```
Sample AMQSCNXC start
Connecting to queue manager MACHINE
using the server connection channel SYSTEM.DEF.SVRCONN
on connection name machine.site.company.com.
Connection established to queue manager MACHINE

Sample AMQSCNXC end
```

Data-Conversion örnek programı

Veri-dönüştürme örnek programı, veri dönüştürme çıkış yordamlarından oluşan bir iskelettir. Veri dönüştürme örneğinin tasarımıyla ilgili bilgi edinin.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Veri dönüştürme örneğinin tasarımı

Her veri dönüştürme çıkış yordamı, adlandırılan tek bir ileti biçimini dönüştürür. Bu iskelet, veri dönüştürme çıkış oluşturma yardımcı programı tarafından oluşturulan kod parçalarına ilişkin bir sarıcı olarak tasarlanmıştır.

Bu yardımcı program her veri yapısı için bir kod parçası üretir; bu tür yapılar bir biçim oluşturur; bu nedenle, tüm biçimdeki veri dönüştürme işlemini yapmak üzere bir yordam üretmek üzere bu iskeletten çok sayıda kod parçası eklenir.

Daha sonra program, dönüştürmenin başarılı mı, yoksa başarısız mı olduğunu denetler ve çağıran için gereken değerleri döndürür.

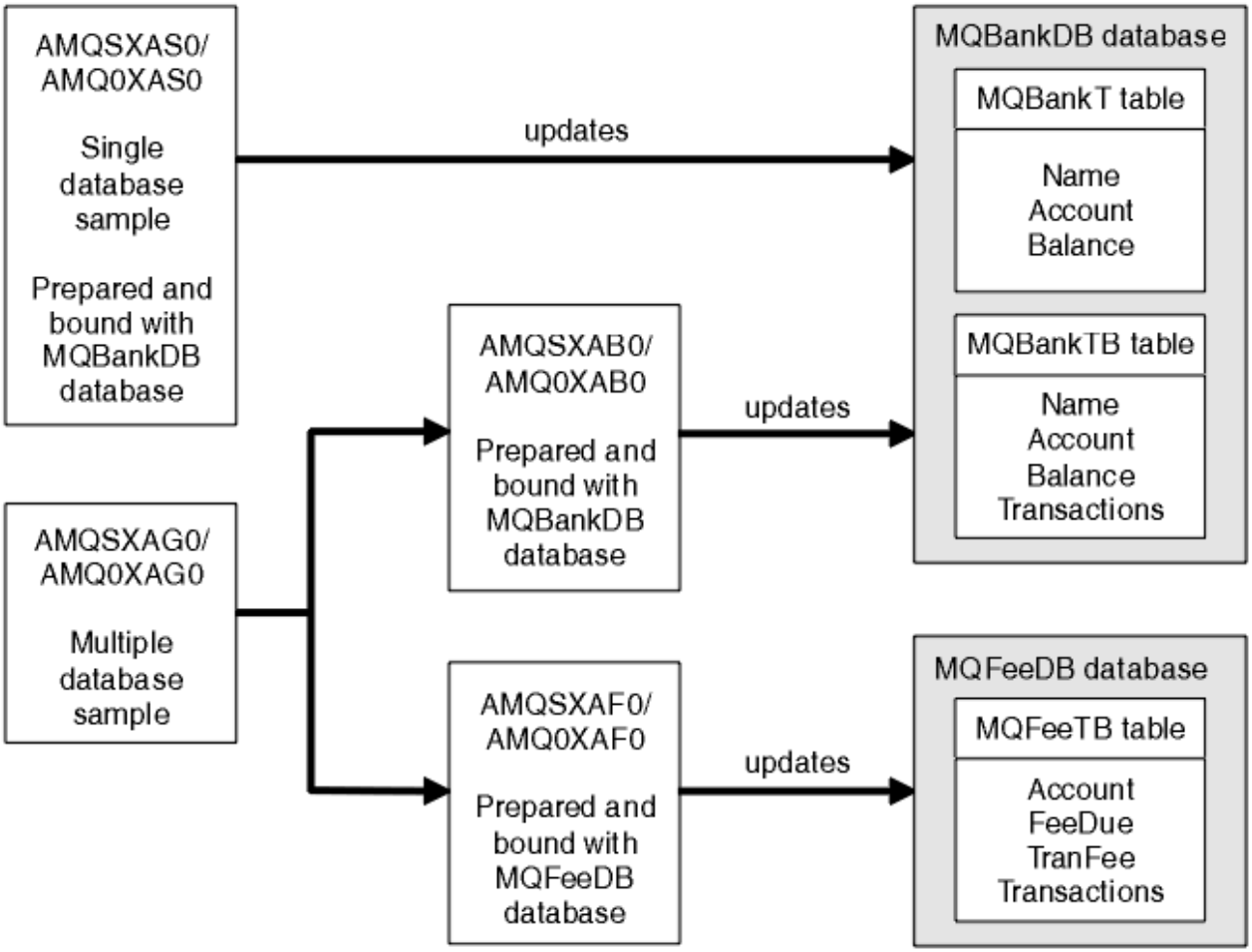
Veritabanı koordinasyonu örnekleri

WebSphere MQ ' in aynı iş birimi içinde hem WebSphere MQ güncellemelerini hem de veritabanı güncellemelerini nasıl koordine edebildiğini gösteren iki örnek verilmiştir.

Bu örnekler şunlardır:

1. AMQSXAS0 (in C) or AMQ0XAS0 (in COBOL), which updates a single database within a WebSphere MQ unit of work.
2. AMQSXAG0 (in C) or AMQ0XAG0 (in COBOL), AMQSXAB0 (in C) or AMQ0XAB0 (in COBOL), and AMQSXAF0 (in C) or AMQ0XAF0 (in COBOL), which together update two databases within a WebSphere MQ unit of work, showing how multiple databases can be accessed. Bu örnekler, MQBEGIN çağrısının, karma SQL ' in ve WebSphere MQ çağrılarının ve nerede ve ne zaman bir veri tabanına bağlanılması için kullanılacağını göstermek için sağlanmıştır.

Şekil 18 sayfa 115 , sağlanan örneklerin veritabanlarını güncellemek için nasıl kullanıldığını gösterir:



Şekil 18. Veritabanı koordinasyonu örnekleri

Programlar bir kuyruktan (syncpoint altında) bir ileti okur, daha sonra, iletiyle ilgili bilgileri kullanarak, ilgili bilgileri veritabanından alıp güncellemesini sağlayın. Daha sonra, veritabanının yeni durumu yazdırılır.

Program mantığı aşağıdaki gibidir:

1. Program bağımsız değişkeninden giriş kuyruğunun adını kullan
2. MQCONN kullanarak varsayılan kuyruk yöneticisine (ya da isteğe bağlı olarak C ' de sağlanan ada) bağlan
3. Hata olmamakla birlikte, giriş için bir kuyruk açın (MQOPEN komutunu kullanın)
4. MQSTART komutunu kullanarak bir iş birimi başlatma
5. Syncpoint altında kuyruktan sonraki iletiyi (MQGET ile) al
6. Veritabanlarından bilgi al
7. Veritabanlarındaki bilgileri güncelle
8. Değişiklikleri MQCMIT kullanarak kesinleştir
9. Güncellenen bilgileri yazdır (hata olarak kullanılabilecek ileti yok ve döngü sona eriyor)
10. MQCLOSE komutunu kullanarak kuyruğu kapat
11. MQDISC komutunu kullanarak kuyruktan bağlantıyı kes

Örneklerde SQL imleçleri kullanılır; bu nedenle, veri tabanlarından (yani birden çok yönetim ortamı) okuyan bir ileti, bir ileti işlenirken kilitlenir ve bu programların birden çok örneğinin aynı anda çalışmasına olanak sağlar. Geçici çizelgeler açık bir şekilde açılmıştır, ancak MQCMIT çağrısıyla örtük olarak kapatılır.

The single database sample (AMQSXAS0 or AMQ0XAS0) has no SQL CONNECT statements and the connection to the database is implicitly made by WebSphere MQ with the MQBEGIN call. Çoklu veritabanı

örneği (AMQSXAG0 ya da AMQ0XAG0, AMQSXAB0 ya da AMQ0XAB0ya da AMQ0XAF0), bazı veritabanı ürünleri yalnızca bir etkin bağlantıya izin verdiğinden, SQL CONNECT deyimlerine sahiptir. Bu durum veritabanı ürününüz için geçerli değilse ya da birden çok veritabanı ürününde tek bir veritabanına erişiyorsanız, SQL CONNECT deyimleri kaldırılabilir.

Örnekler, IBM DB2 veritabanı ürünüyle birlikte hazırlanır; bu nedenle, diğer veritabanı ürünleriyle çalışmak için bunları değiştirmeniz gerekebilir.

The SQL error checking uses routines in UTIL.C and CHECKERR.CBL supplied by DB2. Derleme ve bağlantı oluşturulmadan önce bunlar derlenmeli ya da değiştirilmelidir.

Not: Micro Focus COBOL kaynağı CHECKERR.MFC , program tanıtıcısını büyük harfe çevirmeniz gerekir, bu CHECKERR, AMQ0XAS0 için doğru bağlantı olmalıdır.

Veritabanlarının ve çizelgelerin yaratılması

Örnekleri derlemeden önce veritabanlarını ve tabloları oluşturun.

Veritabanlarını yaratmak için, veritabanı ürününüz için olağan yöntemi kullanın; örneğin:

```
  DB2 CREATE DB MQBankDB
  DB2 CREATE DB MQFeeDB
```

SQL deyimlerini kullanarak çizelgeleri yaratın:

C içinde:

```
EXEC SQL CREATE TABLE MQBankT(Name         VARCHAR(40) NOT NULL,
                              Account      INTEGER     NOT NULL,
                              Balance      INTEGER     NOT NULL,
                              PRIMARY KEY (Account));

EXEC SQL CREATE TABLE MQBankTB(Name         VARCHAR(40) NOT NULL,
                               Account      INTEGER     NOT NULL,
                               Balance      INTEGER     NOT NULL,
                               Transactions INTEGER,
                               PRIMARY KEY (Account));

EXEC SQL CREATE TABLE MQFeeTB(Account      INTEGER     NOT NULL,
                              FeeDue       INTEGER     NOT NULL,
                              TranFee      INTEGER     NOT NULL,
                              Transactions INTEGER,
                              PRIMARY KEY (Account));
```

COBOL 'da:

```
 EXEC SQL CREATE TABLE
   MQBankT(Name      VARCHAR(40) NOT NULL,
           Account   INTEGER     NOT NULL,
           Balance   INTEGER     NOT NULL,
           PRIMARY KEY (Account))
    END-EXEC.

EXEC SQL CREATE TABLE
  MQBankTB(Name      VARCHAR(40) NOT NULL,
           Account   INTEGER     NOT NULL,
           Balance   INTEGER     NOT NULL,
           Transactions INTEGER,
           PRIMARY KEY (Account))
  END-EXEC.

EXEC SQL CREATE TABLE
  MQFeeTB(Account      INTEGER     NOT NULL,
          FeeDue       INTEGER     NOT NULL,
          TranFee      INTEGER     NOT NULL,
          Transactions INTEGER,
          PRIMARY KEY (Account))
  END-EXEC.
```

SQL deyimlerini aşağıdaki gibi kullanarak çizelgelere veri girin:

```
EXEC SQL INSERT INTO MQBankT VALUES ('Mr Fred Bloggs',1,0);
EXEC SQL INSERT INTO MQBankT VALUES ('Mrs S Smith',2,0);
EXEC SQL INSERT INTO MQBankT VALUES ('Ms Mary Brown',3,0);
.
EXEC SQL INSERT INTO MQBankTB VALUES ('Mr Fred Bloggs',1,0,0);
EXEC SQL INSERT INTO MQBankTB VALUES ('Mrs S Smith',2,0,0);
EXEC SQL INSERT INTO MQBankTB VALUES ('Ms Mary Brown',3,0,0);
.
EXEC SQL INSERT INTO MQFeeTB VALUES (1,0,50,0);
EXEC SQL INSERT INTO MQFeeTB VALUES (2,0,50,0);
EXEC SQL INSERT INTO MQFeeTB VALUES (3,0,50,0);
.
```

Not: COBOL için, aynı SQL deyimlerini kullanın, ancak her satırın sonuna END_EXEC ' ı ekleyin.

Örneklerin derlenmesi, derlenmesi ve bağlanması

C ve COBOL içindeki örnekleri derleme öncesi, derleme ve bağlama hakkında bilgi edinin.

.SQC dosyalarını (C içinde) ve .SQB dosyalarını (COBOL ' de) ön derleyin ve .C ya da .CBL dosyalarını üretmek için uygun veritabanına karşı bağ tanımlayın. Bunu yapmak için, veritabanı ürününüz için tipik bir yöntemi kullanın.

C içinde önderleme

```
db2 connect to MQBankDB
db2 prep AMQSXAS0.SQC
db2 connect reset

db2 connect to MQBankDB
db2 prep AMQSXAB0.SQC
db2 connect reset

db2 connect to MQFeeDB
db2 prep AMQSXAF0.SQC
db2 connect reset
```

COBOL ' de ön

```
db2 connect to MQBankDB
db2 prep AMQ0XAS0.SQB bindfile target ibmcob
db2 bind AMQ0XAS0.BND
db2 connect reset

db2 connect to MQBankDB
db2 prep AMQ0XAB0.SQB bindfile target ibmcob
db2 bind AMQ0XAB0.BND
db2 connect reset

db2 connect to MQFeeDB
db2 prep AMQ0XAF0.SQB bindfile target ibmcob
db2 bind AMQ0XAF0.BND
db2 connect reset
```

Derleme ve bağlantı oluşturma

Aşağıdaki örnek komutlar *<DB2TOP>* ve *MQ_INSTALLATION_PATH*simgelerini kullanır. *<DB2TOP>* , DB2 ürününe ilişkin kuruluş dizinini gösterir. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

- AIXüzerinde dizin yolu şu şekilde olur:

  ```
  /usr/lpp/db2_05_00
  ```

- HP-UX ve Solaris üzerinde dizin yolu şöyledir:

```
/opt/IBMdb2/V5.0
```

- Pencereler sistemlerinde, dizin yolu, ürünü kurarken seçilen yola bağlıdır. Varsayılan ayarları seçerseniz, yol şöyle olur:

```
c:\sqllib
```

Not: Before issuing the link command on Pencereler systems, ensure that the LIB environment variable contains paths to the DB2 and WebSphere MQ libraries.

Aşağıdaki dosyaları geçici bir dizine kopyalayın:

- WebSphere MQ kuruluşunuzda bulunan `amqsxag0.c` dosyası

  Not: Bu dosya aşağıdaki dizinlerde bulunabilir:

  – UNIX and Linux sistemlerinde:

  ```
  MQ_INSTALLATION_PATH/samp/xatm
  ```

  – Windows sistemlerinde:

  ```
  MQ_INSTALLATION_PATH\tools\c\samples\xatm
  ```

- `.sqc` kaynak dosyalarını, `amqsxas0.sqc, amqsxaf0.sqc` ve `amqsxab0.sqc`dosyalarını önceden derleyerek edindiğiniz `.c` dosyaları
- The files `util.c` and `util.h` from your DB2 installation.

  Not: Bu dosyalar şu dizinde bulunabilir:

```
<DB2TOP>/samples/c
```

Kullanmakta olduğunuz platforma ilişkin aşağıdaki derleyici komutunu kullanarak her bir `.c` dosyası için nesne dosyalarını oluşturun:

- AIX

```
xlc_r -IMQ_INSTALLATION_PATH/inc -I
<DB2TOP>/include -c -o
 <FILENAME>.o <FILENAME>.c
```

- HP-UX

```
cc -Aa +z -IMQ_INSTALLATION_PATH/inc -I
<DB2TOP>/include -c -o
 <FILENAME>.o <FILENAME>.c
```

- Solaris

```
cc -Aa -KPIC -mt -IMQ_INSTALLATION_PATH
/inc -I<DB2TOP>/include -c -o
 <FILENAME>.o <FILENAME>.c
```

- Windows sistemleri

```
cl /c /IMQ_INSTALLATION_PATH\tools\c\include /I
<DB2TOP>\include
 <FILENAME>.c
```

Kullanmakta olduğunuz platform için aşağıdaki bağlantı komutunu kullanarak amqsxag0 yürütülebilir dosyasını oluşturun:

- AIX

```
xlc_r -H512 -T512 -L<DB2TOP>/lib -ldb2 -LMQ_INSTALLATION_PATH/lib
 -lmqm util.o amqsxaf0.o amqsxab0.o amqsxag0.o -o amqsxag0
```

- HP-UX Revision 11i

```
ld -E -L<DB2TOP>/lib -ldb2 -LMQ_INSTALLATION_PATH/lib -lmqm -lc -lpthread -lcl
 /lib/crt0.o util.o amqsxaf0.o amqsxab0.o amqsxag0.o -o amqsxag0
```

- Solaris

```
cc -mt -L<DB2TOP>/lib -ldb2 -LMQ_INSTALLATION_PATH/lib
 -lmqm -lthread -lsocket -lc -lnsl -ldl util.o
  amqsxaf0.o amqsxab0.o amqsxag0.o -o amqsxag0
```

- Windows sistemleri

```
link util.obj amqsxaf0.obj amqsxab0.obj amqsxag0.obj mqm.lib db2api.lib
 /out:amqsxag0.exe
```

Kullanmakta olduğunuz platforma ilişkin aşağıdaki derleme ve bağlantı komutlarını kullanarak amqsxas0 yürütülebilir dosyasını oluşturun:

- AIX

```
xlc_r -H512 -T512 -L<DB2TOP>/lib -ldb2
 -LMQ_INSTALLATION_PATH/lib -lmqm util.o amqsxas0.o -o amqsxas0
```

- HP-UX Revision 11i

```
ld -E -L<DB2TOP>/lib -ldb2 -LMQ_INSTALLATION_PATH/lib -lmqm -lc -lpthread
 -lcl /lib/crt0.o util.o amqsxas0.o -o amqsxas0
```

- Solaris

```
cc -mt -L<DB2TOP>/lib -ldb2-LMQ_INSTALLATION_PATH/lib
 -lqm -lthread -lsocket -lc -lnsl -ldl util.o
  amqsxas0.o -o amqsxas0
```

- Windows sistemleri

```
link util.obj amqsxas0.obj mqm.lib db2api.lib /out:amqsxas0.exe
```

Ek bilgi

AIX ya da HP-UX üzerinde çalışıyorsanız ve Oracle'a erişmek istiyorsanız, xlc_r derleyicisini kullanın ve libmqm_r.a' ya bağlantı açın.

Örnekleri çalıştırma

C ve COBOL üzerinde veritabanı eşgüdümü örnekleri çalıştırılmadan önce kuyruk yöneticisinin nasıl yapılandırılacağı hakkında bilgi edinmek için bu bilgileri kullanın.

Örnekleri çalıştırmadan önce, kuyruk yöneticisini kullanmakta olduğunuz veritabanı ürünüyle yapılandırın. Bunun nasıl yapacağına ilişkin bilgi için bkz. “Senaryo 1: Kuyruk yöneticisi koordinasyonu gerçekleştirir” sayfa 41.

Aşağıdaki başlıklar, C ve COBOL ' de örnekleri çalıştırma hakkında bilgi sağlar:

- “C örnekleri” sayfa 120
- “COBOL örnekleri” sayfa 120

C örnekleri

İletiler kuyruktan okunmak üzere aşağıdaki biçimde olmalıdır:

```
UPDATE Balance change=nnn WHERE Account=nnn
```

İletileri kuyruğa koymak için MQSPUT kullanılabilir.

Veritabanı koordinasyonu örnekleri iki parametre alır:

1. Kuyruk adı (gerekli)
2. Kuyruk yöneticisi adı (isteğe bağlı)

Assuming that you have created and configured a queue manager for the single database sample called singDBQM, with a queue called singDBQ, you increment Mr Fred Bloggs's account by 50 as follows:

```
AMQSPUT singDBQ singDBQM
```

Daha sonra, aşağıdaki iletiyle ilgili anahtar:

```
UPDATE Balance change=50 WHERE Account=1
```

Kuyruğa birden çok ileti yerleştirebilirsiniz.

```
AMQSXAS0 singDBQ singDBQM
```

Daha sonra, Bay Fred Bloggs 'in hesabında güncellenen durum yazdırılır.

Assuming that you have created and configured a queue manager for the multiple-database sample called multDBQM, with a queue called multDBQ, you decrement Ms Mary Brown's account by 75 as follows:

```
AMQSPUT multDBQ multDBQM
```

Daha sonra, aşağıdaki iletiyle ilgili anahtar:

```
UPDATE Balance change=-75 WHERE Account=3
```

Kuyruğa birden çok ileti yerleştirebilirsiniz.

```
AMQSXAG0 multDBQ multDBQM
```

Daha sonra, Bayan Mary Brown 'ın hesabında güncellenen durum yazdırılır.

COBOL örnekleri

İletiler kuyruktan okunmak üzere aşağıdaki biçimde olmalıdır:

```
UPDATE Balance change=snnnnnnnn WHERE Account=nnnnnnnn
```

Basitlik için, `Balance change` imzalı sekiz karaktmeli bir sayı olmalı ve `Account` sekiz karakterden oluşan bir sayı olmalıdır.

İletileri kuyruğa koymak için örnek AMQSPUT örneği kullanılabilir.

Örnekler parametre alır ve varsayılan kuyruk yöneticisini kullanır. Bu, herhangi bir zamanda örneklerden yalnızca birini çalıştırabilecek şekilde yapılandırılabilir. Assuming that you have configured the default queue manager for the single database sample, with a queue called singDBQ, you increment Mr Fred Bloggs's account by 50 as follows:

```
AMQSPUT singDBQ
```

Daha sonra, aşağıdaki iletiyle ilgili anahtar:

```
UPDATE Balance change=+00000050 WHERE Account=00000001
```

Kuyruğa birden çok ileti koyabilirsiniz:

```
AMQ0XAS0
```

Kuyruğun adını yazın:

```
singDBQ
```

Daha sonra, Bay Fred Bloggs 'in hesabında güncellenen durum yazdırılır.

Assuming that you have configured the default queue manager for the multiple database sample, with a queue called multDBQ, you decrement Ms Mary Brown's account by 75 as follows:

```
AMQSPUT multDBQ
```

Daha sonra, aşağıdaki iletiyle ilgili anahtar:

```
UPDATE Balance change=-00000075 WHERE Account=00000003
```

Kuyruğa birden çok ileti koyabilirsiniz:

```
AMQ0XAG0
```

Kuyruğun adını yazın:

```
multDBQ
```

Daha sonra, Bayan Mary Brown 'ın hesabında güncellenen durum yazdırılır.

Ölü-harfli kuyruk işleyicisi örneği

Örnek bir dead-letter queue işleyicisi sağlanıyor, yürütülebilir sürümün adı amqsdlq. RUNMQDLQ ' dan farklı bir ileti kuyruğu işleyicisi istiyorsanız, tabanınız olarak kullanmak üzere, örneğin kaynağı kullanılabilir.

Örnek, ürün içinde sağlanan ölü harf işleyiciye benzer, ancak izleme ve hata raporlamasının farklı olduğunu da sağlar. Kullanabileceğiniz iki ortam değişkeni vardır:

ODQ_TRACE
İzlemeyi değiştirmek için EVET ya da Evet olarak ayarla

ODQ_MSG
Hata ve bilgi iletilerini içeren dosyanın adını girin. Sağlanan dosyaya amqsdlq.msgadı verilir.

Platformunuza bağlı olarak, **export** ya da **set** komutlarını kullanarak ortamınız tarafından bilinen bu değişkenleri **unset** komutunu kullanarak kapatmanız gerekir.

You can modify the error message file, amqsdlq.msg, to suit your own requirements. The sample puts messages to stdout, ***değil*** to the WebSphere MQ error log file.

Altyapınıza ilişkin Yönetme ya da *System Management Guide* adlı kılavuz, ölü harf işleyicinin nasıl çalıştığını ve nasıl çalıştıdığınızı açıklar.

Dağıtım Listesi örnek programı

Dağıtım Listesi örneği amqsptl0 , ileti kuyruklarına bir ileti yerleştirmenin bir örneğini verir. MQPUT örneğine, amqsput0' a dayalıdır.

Dağıtım Listesi örneği çalıştırılıyor, amqsptl0

Dağıtım Listesi örneği, Koyma örneklerine benzer bir şekilde çalışır.

Bu değiştirge aşağıdaki değiştirgeleri alır:

- Kuyrukların adları
- Kuyruk yöneticilerinin adları

Bu değerler çift olarak girilir. Örneğin:

```
amqsptl0 queue1 qmanagername1 queue2 qmanagername2
```

Kuyruklar, MQPUT kullanılarak kuyruklara açılır ve MQPUT ile kuyruklara konalır. Kuyruk ya da kuyruk yöneticisi adlarından herhangi biri tanınmazsa neden kodları döndürülür.

İletiler arasında akış yapabilmeleri için kuyruk yöneticileri arasında kanallar tanımlamayı unutmayın. Örnek program bunu sizin için yapmaz.

Dağıtım Listesi örneğinin tasarımı

İleti Kayıtları 'nı (MQPMR ' lar) her hedef için ileti özniteliklerini belirtin. Örnek, *MsgId* ve *CorrelId*için değerler sağlar ve bu değerler MQMD yapısında belirtilen değerleri geçersiz kılar.

MQPMO yapısındaki *PutMsgRecFields* alanı, MQPMRS ' de hangi alanların bulunduğunu gösterir:

```
  MQLONG PutMsgRecFields=MQPMRF_MSG_ID + MQPMRF_CORREL_ID;
```

Daha sonra, örnek yanıt kayıtlarını ve nesne kayıtlarını ayırır. Nesne kayıtları (MQORs) en az bir çift ad ve çift sayıda ad gerektirir; bu da, *ObjectName* ve *ObjectQMgrName*.

Sonraki aşama, MQCONN kullanan kuyruk yöneticilerine bağlanmayı içerir. Bu örnek, MQOR içindeki ilk kuyrukla ilişkilendirilmiş kuyruk yöneticisine bağlanmayı dener; bu işlem başarısız olursa, nesne kayıtlarında sırayla devam eder. Herhangi bir kuyruk yöneticisine ve program çıkışlarına bağlanmak olanaklı değilse, size bilgi verilirir.

Hedef kuyruklar, MQPUT kullanılarak açılır ve MQPUT kullanılarak bu kuyruklara ileti konması gerekir. Yanıt kayıtlarında (MQRR ' lar) herhangi bir sorun ve hata bildirilir.

Son olarak, hedef kuyruklar MQCLOSE kullanılarak kapatılır ve program MQDISC kullanılarak kuyruk yöneticisinden bağlantıyı keser. *CompCode* ve *Reason*' yi belirten her çağrı için aynı yanıt kayıtları kullanılır.

Echo örnek programları

Echo örnek programları, ileti kuyruğundan yanıt kuyruğuna bir ileti echo eder.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Programlar, tetiklenen programlar olarak çalıştırılmak üzere tasarlanmıştır.

UNIX, Linux ve Windows sistemlerinde, bunların tek girişi, hedef kuyruğun adını ve kuyruk yöneticisini içeren bir MQTMC2 (tetikleme iletisi) yapısıdır. COBOL sürümü, varsayılan kuyruk yöneticisini kullanır.

When you have set the definition correctly, first start AMQSERV4 in one job, then start AMQSREQ4 in another. AMQSERV4yerine AMQSTRG4 ' yi kullanabilirsiniz, ancak olası iş gönderimi gecikmeleri, gerçekleşenleri takip etmeyi daha az kolaylaştırabilirdi.

Use the Request sample programs to send messages to queue SYSTEM.SAMPLE.ECHO. Echo örnek programları, istek iletisinde, istek iletisinde belirtilen yanıt kuyruğuna veri içeren bir yanıt iletisi gönderir.

Echo örnek programlarının tasarımı

Program, tetikleme iletisi yapısında adı belirtilen kuyruğu açarken, bu işlem başlatıldığında iletileceği bir kuyrukta yer alan kuyruğu açar. (For clarity, we will call this the *istek kuyruğu*.) Program, bu kuyruğu paylaşılan giriş için açmak için MQOPEN çağrısını kullanır.

Program, bu kuyruktan iletileri kaldırmak için MQGET çağrısını kullanır. Bu çağrı, 5 saniye bekleme süresi ile MQGMO_ACCEPT_TRUNCATED_MSG, MQGMO_CONVERT ve MQGMO_WAWT seçeneklerini kullanır. Program, bir istek iletisi olup olmadığını görmek için her iletinin tanımlayıcısını sınar; değilse, program iletiyi atar ve bir uyarı iletisi görüntüler.

Her giriş satırı için program, metni bir arabelleğe okur ve MQPUT1 çağrısını kullanarak, o satırın metnini içeren bir istek iletisini yanıtlama kuyruğunda kullanır.

MQGET çağrısının başarısız olması durumunda, program yanıt kuyruğuna bir rapor iletisi koyar ve ileti tanımlayıcısının *Feedback* alanını MQGET tarafından döndürülen neden koduna göre ayarlar.

İstek kuyruğunda bir ileti kalmadığında, program o kuyruğu kapatır ve kuyruk yöneticisinden bağlantıyı keser.

Get Sample programlar

Alma örnek programları, MQGET çağrısını kullanarak kuyruktan ileti alır.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Get Sample programının tasarımı

Program, MQOPEN çağrısını MQOO_INPUT_AS_Q_DEF seçeneğiyle kullanarak hedef kuyruğu açar. Kuyruğu açamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler.

Kuyrukta yer alan her ileti için, program iletiyi kuyruktan kaldırmak için MQGET çağrısını kullanır, daha sonra iletide bulunan verileri görüntüler. MQGET çağrısı, kuyruğun üzerinde bir ileti yoksa programın bu dönemi bekleyeceği şekilde, 15 saniyelik bir *WaitInterval* belirten MQGMO_WATM seçeneğini kullanır. Bu aralığın süresi dolmadan bir ileti gelmezse, çağrı başarısız olur ve MQRC_NO_MSG_AVAILEABLE neden kodunu döndürür.

Bu program, bu alanları, aldığı iletide yer alan değerlere ayarlaması nedeniyle, her MQGET çağrısından sonra MQMD yapısının *MsgId* ve *CorrelId* alanlarını nasıl temizlemeniz gerektiğini gösterir. Bu alanların temizlenmesi, art arda gelen MQGET çağrılarının iletilerin kuyrukta tutulmakta olduğu sırayla alma çağrıları anlamına gelir.

MQGET çağrısı, sabit büyüklerin arabelleğinden birini belirtir. Bu arabellekten daha uzun bir ileti varsa, arama başarısız olur ve program durur.

Bu program, MQGET çağrısına ilişkin MQRC_NO_MSG_AVAILEABLE neden kodunu döndürünceye ya da MQGET çağrısının başarısız olduğu zamana kadar devam eder. Arama başarısız olursa, program neden kodunu içeren bir hata iletisi görüntüler.

Daha sonra, program MQCLOSE çağrısını kullanarak kuyruğu kapatır.

amqsget ve amqsgetc örneklerinin çalıştırılması

Bu programlar her biri iki parametre alır:

1. Kaynak kuyruğun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)

Bir kuyruk yöneticisi belirtilmezse, amqsget varsayılan kuyruk yöneticisine bağlanır ve amqsgetc, bir ortam değişkeniyle ya da istemci kanal tanımlama dosyası tarafından tanımlanan kuyruk yöneticisine bağlanır.

Bu programları çalıştırmak için aşağıdakilerden birini girin:

- `amqsget myqueue qmanagername`
- `amqsgetc myqueue qmanagername`

Burada `myqueue` , programın iletileri alacağı kuyruğun adıdır, `qmanagername` ise `myqueue`' un sahibi olan kuyruk yöneticidir.

`qmanagername`öğesini çıkarırsanız, programlar varsayılan olarak ya da MQI istemcisi durumunda, bir ortam değişkeniyle ya da istemci kanal tanımlama dosyası tarafından tanımlanan kuyruk yöneticisini varsayar.

Yüksek kullanılabilirlikli örnek programlar

amqsghac, **amqsphac**ve **amqsmhac** yüksek düzeyde kullanılabilirlik örnek programları, bir kuyruk yöneticisinin arızalanması sonrasında kurtarmayı göstermek için otomatik istemci yeniden bağlantısını kullanır. **amqsfhac** , ağ üzerinden depolama kullanan bir kuyruk yöneticisinin bir hatanın ardından veri bütünlüğünü korumasını denetler.

amqsghac, **amqsphac**ve **amqsmhac** programları komut satırından başlatılır ve çok eşgörünümlü bir kuyruk yöneticisinin bir eşgörünümü başarısız olduktan sonra yeniden bağlantıyı göstermek için birlikte kullanılabilir.

Diğer bir yöntem olarak, genellikle bir kuyruk yöneticisi grubunda yapılandırılmış olan tek eşgörünüm kuyruğu yöneticilerine istemci yeniden bağlantısını göstermek için **amqsghac**, **amqsphac**ve **amqsmhac** örneklerini de kullanabilirsiniz.

Örneğin kolay yapılandırılabilmesi için, örnek programların başlatılmış, durdurulmuş ve yeniden başlatılmış tek bir yönetim ortamı kuyruk yöneticisine yeniden bağlanması gösterilir; bkz. “Kuyruk yöneticisinin ayarlanması ve denetlenmesi” sayfa 126.

Dosya sistemi bütünlüğünü denetlemek için **amqmfsck** ile paralel olarak **amqsfhac** komutunu kullanın. Daha fazla bilgi için bkz. **amqmfsck** (dosya sistemi denetimi) ve Paylaşılan dosya sistemi davranışının doğrulanması .

amqsphac *queueName* [*qMgrAd*]

- **amqsphac** bir IBM WebSphere MQ MQI client uygulamasıdır. Her ileti arasında iki saniyelik bir gecikme ile bir kuyruğa ileti dizisi koyar ve olay işleyicisine gönderilen olayları görüntüler.
- İletileri kuyruğa koymak için eşitleme noktası kullanılmaz.
- Aynı kuyruk yöneticisi grubundaki herhangi bir kuyruk yöneticisiyle yeniden bağlantı kurulabilir.

amqsghac *queueName* [*qMgrAd*]

- **amqsghac** bir IBM WebSphere MQ MQI client uygulamasıdır. Bir kuyruktan ileti alır ve olay işleyicisine gönderilen olayları görüntüler.
- Kuyruktan ileti almak için eşitleme noktası kullanılmaz.
- Aynı kuyruk yöneticisi grubundaki herhangi bir kuyruk yöneticisiyle yeniden bağlantı kurulabilir.

amqsmhac -s *sourceQueueAdı* -t *targetQueueAdı* [-m *qMgrAdı*] [-w *waitInterval*]

- **amqsmhac** bir IBM WebSphere MQ MQI client uygulamasıdır. Program tamamlanmadan önce alınan son iletiden 15 dakika sonra varsayılan bekleme aralığıyla iletileri bir kuyruktan diğerine kopyalar.
- İletiler eşitleme noktası içinde kopyalanır.

- Yalnızca aynı kuyruk yöneticisiyle yeniden bağlantı kurulabilir.

amqsfhac *QueueManagerAdı QueueName SideQueueAdı InTransactionCount RepeatCount* (0|1|2)

- **amqsfhac** bir IBM WebSphere MQ MQI client uygulamasıdır. NAS ya da küme kütük sistemi gibi ağa bağlı depolama kullanan bir IBM WebSphere MQ çok eşgörünümlü kuyruk yöneticisinin veri bütünlüğünü korumasını denetler. **amqsfhac** komutunu Paylaşılan dosya sistemi davranışının doğrulanması içinde çalıştırmak için aşağıdaki adımları izleyin.
- *QueueManagerAd*olanağına bağlanırken MQCNO_RECONNECT_Q_MGR seçeneğini kullanır. Kuyruk yöneticisi başarısız olduğunda otomatik olarak yeniden bağlanır.
- *InTransactionCount (Sayı)*⋆*RepeatCount* kalıcı iletilerini *QueueName* ' e koyar; bu sırada kuyruk yöneticisi birçok kez başarısız olur. **amqsfhac** her seferinde kuyruk yöneticisine yeniden bağlanır ve devam eder. Sınama, hiçbir iletinin kaybolmadığından emin olmaktır.
- *InTransactionCount* (İşlem Sayısı) iletileri her işlem içine konmuştur. İşlem *RepeatCount* kez yinelenir. Bir hareket içinde bir hata oluşursa, **amqsfhac** kuyruk yöneticisine yeniden bağlandığında **amqsfhac** hareketi geriye işleyip yeniden sunar.
- Ayrıca, iletileri *SideQueueAd*' a da yerleştirir. Tüm iletilerin *QueueName* kuyruğundan başarıyla kesinleştirilip kesinleştirilmediğini ya da geriye işlenip işlenmediğini denetlemek için *SideQueueName* komutunu kullanır. Bir tutarsızlık saptarsa, bir hata iletisi yazar.
- Son parametreyi (0|1|2) olarak ayarlayarak çıktı izleme miktarını **amqsfhac** değerine ayarlayın.

  0
  En azından çıktı.

  1
  Orta çıkış.

  2
  Çoğu çıktı.

İstemci bağlantısı yapılandırılması

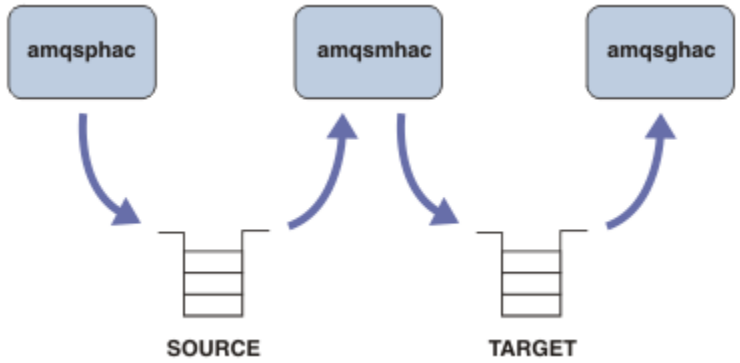
Örnekleri çalıştırmak için bir istemci ve sunucu bağlantı kanalı yapılandırmanız gerekir. İstemci doğrulama yordamı, bir istemci test ortamının nasıl ayarlanacağını açıklar. Bkz. İstemci kuruluşunun doğrulanması.

Diğer bir seçenek olarak, aşağıdaki örnekte sağlanan yapılanışı kullanın.

amqsghac, amqsphacve amqsmhac Yöntemlerinin Kullanımı-Örnek

Bu örnek, tek bir yönetim ortamı kuyruk yöneticisini kullanarak yeniden bağlanabilir istemcileri gösterir.

İletiler, **amqsphac**tarafından SOURCE kuyruğa yerleştirilir, **amqsmhac**tarafından TARGET 'e aktarılır ve **amqsghac**tarafından TARGET ' den alınır; bkz. Şekil 19 sayfa 125.



Şekil 19. Yeniden bağlanabilir istemci örnekleri

Örnekleri çalıştırmak için aşağıdaki adımları izleyin.

1. Aşağıdaki komutları içeren bir `hasamples.tst` dosyası yaratın:

```
DEFINE QLOCAL(SOURCE) REPLACE
DEFINE QLOCAL(TARGET) REPLACE
DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN)   TRPTYPE(TCP) +
       MCAUSER(MUSR_MQADMIN) REPLACE
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) +
       CONNAME('LOCALHOST(2345)') QMNAME(QM1) REPLACE
ALTER  LISTENER(SYSTEM.DEFAULT.LISTENER.TCP) TRPTYPE(TCP) +
       PORT(2345)
START  LISTENER(SYSTEM.DEFAULT.LISTENER.TCP)
START  CHANNEL(CHANNEL1)
```

2. Bir komut istemine aşağıdaki komutları yazın:
   a. `crtmqm QM1`
   b. `strmqm QM1`
   c. `runmqsc QM1 < hasamples.tst`
3. **MQCHLLIB** ortam değişkenini `AMQCLCHL.TAB` istemci kanal tanımlama dosyasının yoluna ayarlayın; örneğin, `SET MQCHLLIB=C:\IBM\MQ\MQ7\Data\qmgrs\QM1\@ipcc`.
4. **MQCHLLIB** kümesiyle üç yeni pencere açın; örneğin, Windows' ta, önceki komut istemine **start** yazın ve her bir programı pencerelerden birinde başlatın. “Kuyruk yöneticisinin ayarlanması ve denetlenmesi” sayfa 126içindeki “5” sayfa 127 adımına bakın.)
5. Kuyruk yöneticisini durdurmak için `endmqm -r -p QM1` komutunu yazın ve istemcilerin yeniden bağlanmasına izin verin.
6. Kuyruk yöneticisini yeniden başlatmak için `strmqm QM1` komutunu yazın.

amqsghac, **amqsphac**ve **amqsmhac** örneklerini Windows üzerinde çalıştırmanın sonuçları aşağıdaki örneklerde gösterilmiştir.

Kuyruk yöneticisinin ayarlanması ve denetlenmesi

1. Kuyruk yöneticisini yaratın.

```
C:\>crtmqm QM1
WebSphere MQ queue manager created.
Directory 'C:\IBM\MQ\MQ7\Data\qmgrs\QM1' created.
Creating or replacing default objects for QM1.
Default objects statistics : 67 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

   MQCHLLIB değişkenini daha sonra ayarlamak için veri dizinini hatırlayın.
2. Kuyruk yöneticisini başlatın.

```
C:\>strmqm QM1
WebSphere MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
WebSphere MQ queue manager 'QM1' started.
```

3. Kuyrukları ve kanalları oluşturun, dinleyici kapısını değiştirin ve dinleyiciyi ve kanalı başlatın.

```
C:\>runmqsc QM1 < hasamples.tst
5724-H72 (C) Copyright IBM Corp. 1994, 2024. ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.

     1 : DEFINE QLOCAL(SOURCE) REPLACE
AMQ8006: WebSphere MQ queue created.
     2 : DEFINE QLOCAL(TARGET) REPLACE
AMQ8006: WebSphere MQ queue created.
     3 : DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN)   TRPTYPE(TCP) MCAUSER(MUSR_MQADMIN)
REPLACE
AMQ8014: WebSphere MQ channel created.
     4 : DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN)  TRPTYPE(TCP) CONNAME('LOCALHOST(2345)')
```

```
QMNAME(QM1) REPLACE
AMQ8014: WebSphere MQ channel created.
     5 : ALTER LISTENER(SYSTEM.DEFAULT.LISTENER.TCP) TRPTYPE(TCP) PORT(2345)
AMQ8623: WebSphere MQ listener changed.
     6 : START LISTENER(SYSTEM.DEFAULT.LISTENER.TCP)
AMQ8021: Request to start WebSphere MQ Listener accepted.
     7 : START CHANNEL(CHANNEL1)
AMQ8018: Start WebSphere MQ channel accepted.
7 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

4. İstemci kanal çizelgesini istemcilere bildir.

   "1" sayfa 126. adımda **crtmqm** komutundan döndürülen veri dizinini kullanın ve **MQCHLLIB** değişkenini ayarlamak için `@ipcc` dizinini ekleyin.

```
C:\>SET MQCHLLIB=C:\IBM\MQ\MQ7\Data\qmgrs\QM1\@ipcc
```

5. Diğer pencerelerdeki örnek programları başlat

```
C:\>start amqsphac SOURCE QM1
C:\>start amqsmhac -s SOURCE -t TARGET -m QM1
C:\>start amqsghac TARGET QM1
```

6. Kuyruk yöneticisini sona erdirin ve yeniden başlatın.

```
C:\>endmqm -r -p QM1
Waiting for queue manager 'QM1' to end.
WebSphere MQ queue manager 'QM1' ending.
WebSphere MQ queue manager 'QM1' ended.

C:\>strmqm QM1
WebSphere MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
WebSphere MQ queue manager 'QM1' started.
```

amqsphac

```
Sample AMQSPHAC start
target queue is SOURCE
message <Message 1>
message <Message 2>
16:25:22 : EVENT : Connection Reconnecting (Delay: 0ms)
16:25:45 : EVENT : Connection Reconnecting (Delay: 0ms)
16:26:02 : EVENT : Connection Reconnectedmessage
<Message 3>
message <Message 4>
message <Message 5>
```

amqsmhac

```
Sample AMQSMHA0 start
16:25:22 : EVENT : Connection Reconnecting (Delay: 0ms)
16:25:45 : EVENT : Connection Reconnecting (Delay: 0ms)
16:26:02 : EVENT : Connection Reconnected
No more messages.
Sample AMQSMHA0 end
C:\>
```

amqsghac

```
Sample AMQSGHAC start
message <Message 1>
message <Message 2>
16:25:22 : EVENT : Connection Reconnecting (Delay: 0ms)
16:25:45 : EVENT : Connection Reconnecting (Delay: 0ms)
```

```
16:26:02 : EVENT : Connection Reconnected
message <Message 3>
message <Message 4>
message <Message 5>
```

İlgili görevler

Paylaşılan dosya sistemi davranışının doğrulanması

İlgili başvurular

amqmfsck (dosya sistemi denetimi)

Sorma örnek programları

Sorgula örnek programları, MQINQ çağrısını kullanan bir kuyruğun bazı özniteliklerine ilişkin bilgi içerir.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Bu programların tetiklenen programlar olarak çalıştırılması amaçlanır, bu nedenle bunların tek girişi, IBM i, Windows, UNIX and Linux sistemleri için bir MQTMC2 (tetikleme iletisi) yapısıdır. Bu yapı, sorgulanacak özniteliklere sahip bir hedef kuyruğun adını içerir. C sürümü kuyruk yöneticisi adını da kullanır. COBOL sürümü, varsayılan kuyruk yöneticisini kullanır.

For the triggering process to work, ensure that the Inquire sample program that you want to use is triggered by messages arriving on queue SYSTEM.SAMPLE.INQ. To do this, specify the name of the Inquire sample program that you want to use in the *ApplicId* field of the process definition SYSTEM.SAMPLE.INQPROCESS. Örnek kuyruğun tetikleme tipi FIRST; istek örneğini çalıştırmadan önce kuyruğunda önceden ileti varsa, gönderdiğiniz iletiler sorgulamak için tetikleme örneği tetiklenmez.

Tanımlamayı doğru bir şekilde ayarladığınızda:

- UNIX, Linux ve Windows sistemleri için tek bir oturumda **runmqtrm** programını başlatın ve ardından başka bir oturumda `amqsreq` programını başlatın.

İstek örnek programlarını kullanarak, her biri yalnızca bir kuyruk adı içeren istek iletilerini SYSTEM.SAMPLE.INQ. Her istek iletisi için, Sorgula örnek programları, istek iletisinde belirtilen kuyruğa ilişkin bilgileri içeren bir yanıt iletisi gönderir. Yanıtlar, istek iletisinde belirtilen yanıtlama kuyruğuna gönderilir.

Sorgulamak için örnek program tasarımı

Program, tetikleme iletisi yapısında adı belirtilen kuyruğu açarken, bu işlem başlatıldığında iletileceği bir kuyrukta yer alan kuyruğu açar. (For clarity, we will call this the *istek kuyruğu*.) Program, bu kuyruğu paylaşılan giriş için açmak için MQOPEN çağrısını kullanır.

Program, bu kuyruktan iletileri kaldırmak için MQGET çağrısını kullanır. Bu çağrı, 5 saniye bekleme süresi ile MQGMO_ACCEPT_TRUNCATED_MSG ve MQGMO_WADET seçeneklerini kullanır. Program, bir istek iletisi olup olmadığını görmek için her iletinin tanımlayıcısını sınar; değilse, program iletiyi atar ve bir uyarı iletisi görüntüler.

İstek kuyruğundan kaldırılan her istek iletisi için, program, verilerde bulunan kuyruğun adını (*hedef kuyruğu*çağıracak) okur ve MQOO_INQ seçeneğiyle MQOPER çağrısını kullanarak kuyruğu açar. Daha sonra, program, hedef kuyruğun *InhibitGet*, *CurrentQDepth*ve *OpenInputCount* özniteliklerinin değerlerini sorgulamak için MQINQ çağrısını kullanır.

MQINQ çağrısı başarılı olursa, program yanıt kuyruğuna yanıt iletisi koymak için MQPUT1 çağrısını kullanır. Bu ileti, üç özniteliğin değerlerini içerir.

MQAUT ya da MQINQ çağrısı başarısız olursa, program yanıt kuyruğuna bir rapor iletisi koymak için MQPUT1 çağrısını kullanır. Bu rapor iletisinin ileti tanımlayıcısının *Feedback* alanında, başarısız olan buna bağlı olarak, MQOPED ya da MQINQ çağrısının döndürdüğü neden kodudur.

MQINQ çağrısından sonra program, MQCLOSE çağrısını kullanarak hedef kuyruğu kapar.

İstek kuyruğunda bir ileti kalmadığında, program o kuyruğu kapatır ve kuyruk yöneticisinden bağlantıyı keser.

Message Handle örnek programının Sorgusu Özellikleri

AMQSIQMA, bir ileti kuyruğunun özelliklerini bir ileti kuyruğundan sorgulamak için kullanılan bir örnek C programıdır ve MQINQMP API çağrısının kullanılmasına bir örnektir.

Bu örnek, bir ileti tanıtıcısı yaratır ve bunu, MQGMO yapısının MsgHandle alanına yerleştirir. Daha sonra, örnek bir ileti alır ve ileti tanıtıcısının doldurulduğu tüm özellikleri sorgular ve yazdırır.

```
C:\Program Files\IBM\WebSphere MQ\tools\c\Samples\Bin >amqsiqm Q QM1
Sample AMQSIQMA start
property name <MyProp> value <MyValue>
message text <Hello world!>
Sample AMQSIQMA end
```

Yayınlama/Abone Olma örnek programları

The publish/subscribe sample programs demonstrate the use of the publish and subscribe features in WebSphere MQ.

WebSphere MQ yayınlama/abone olma arabirimine nasıl programlaşacağını gösteren üç adet C dili örnek programı vardır. Eski arabirimleri kullanan bazı C örnekleri vardır ve Java örnekleri vardır. Java örnekleri, com.ibm.mq.jariçindeki WebSphere MQ yayınlama/abone olma arabirimini ve com.ibm.mqjmsiçindeki JMS yayınlama/abone olma arabirimlerini kullanır. JMS örnekleri bu konuda kapsanmıyor.

C

Find the publisher sample `amqspub` in the C samples folder. Bunu, ilk parametre olarak beğendiğiniz herhangi bir konu adıyla çalıştırın ve ardından isteğe bağlı bir kuyruk yöneticisi adını girin. Örneğin, `amqspub mytopic QM3` . Ayrıca, `amqspubcad`lı bir istemci sürümü de vardır. İstemci sürümünü çalıştırmayı seçerseniz, ayrıntılar için önce “Örnek programların hazırlanması ve çalıştırılması” sayfa 104 ' i (bkz..) bakın.

Yayınlayıcı, varsayılan kuyruk yöneticisine bağlanır ve çıkışa yanıt verir: `target topic is mytopic` . Bu pencereye şu andan itibaren girdiğiniz her satır, `mytopic`olarak yayınlanır.

Aynı dizinde başka bir komut penceresi açın ve aynı konu adını ve isteğe bağlı kuyruk yöneticisi adını belirterek abone programını `amqssub`çalıştırın. Örneğin, `amqssub mytopic QM3`.

Abone çıkışa yanıt veriyor, `Calling MQGET : 30 seconds wait time`. Şu andan itibaren, yayınlayıcıya yazdığınız satırlar abonenin çıkışında görünür.

Başka bir komut penceresinde başka bir abone başlatın ve her iki aboneyi de yayın olarak izleyin.

Ayar seçenekleri de içinde olmak üzere parametrelere ilişkin eksiksiz belgeler için örnek kaynak koduna bakın. Abone seçenekleri alanına ilişkin değerler şu konuda açıklanmıştır: Options (MQUZE).

Komut satırı anahtarları olarak ek abonelik seçenekleri sunan başka birsubscriberabone örneği `amqssbx`vardır.

Abone sona erdirildikten sonra alıkonan dayanıklı abonelikler kullanarak aboneyi çağırmak için `amqssbx -d mysub -t mytopic -k` yazın.

Yayınlayıcıyı kullanarak başka bir öğe yayınlayarak aboneliği test edin. Abonenin sona ermesi için 30 saniye bekleyin. Aynı konu altında birkaç öğe daha yayınlayın. Aboneyi yeniden başlatın. Abone çalışmazken yayınlanan son öğe, abone tarafından görüntülenerek hemen yeniden başlatılır.

C mirası

Kuyruğa alınan komutları gösteren ek bir C örneği kümesi vardır. Bu örneklerden bazıları başlangıçta MQ0C Supportpac 'ın bir parçası olarak gönderilmişti. Uyumluluk nedenleri için, örneklerin tam olarak desteklendiği yeteneklerdir.

Kuyruklanan komut arabirimini kullanmaktan vazgeçmenizi öneriyoruz. Yayınlama/abone olma API 'sinden çok daha karmaşıktır ve karmaşık kuyruğa alınmış komutları programlamak için zorlayıcı bir işlev

nedeni yoktur. Ancak, kuyruğa yollanan yaklaşımı daha uygun bulabilir, belki de arayüzü kullanmakta olduğunuz için ya da programlama ortamınız karmaşık bir ileti oluşturmanızı ve MQSUB ' ye farklı çağrılar oluşturmak yerine soysal bir MQPUT adını koymanızı kolaylaştırır.

Ek örnekler, `samples` klasöründeki pubsub alt dizininde bulunur.

Çizelge 20 sayfa 130içinde listelenen altı tür örnek vardır.

Çizelge 20. Eski yayınlama/abone olma örnek C programlarının kategorileri

Kategori	Programlar	Açıklamalar
RFH1	`amqssr1a.c` `amqspr1a.c`	Simple publish/subscribe example built using RFH1 format messages.
RFH2	`amqssr2a.c` `amqspr2a.c`	Simple publish/subscribe example built using RFH2 format messages.
MQAI örnekleri	`amqsppca.c` `amqsspca.c`	PCF komutları ve MQAI komut arabirimi kullanılarak oluşturulan basit yayınlama/abone olma örneği.
MA0C Results service using RFH1	`amqsgama.c` `amqsresa.c`	Results service built using RFH1 headers 1. Requires the queues defined in `amqsgama.tst` and `amqsresa.tst` 2. `amqsresa` must be started before `amqsgama`
MA0C RFH2kullanılarak sonuç hizmeti	`amqsgr2a.c` `amqsrr2a.c`	Results service built using RFH2 headers 1. Requires the queues defined in `amqsgama.tst` and `amqsresa.tst` 2. `amqsresa` must be started before `amqsgama`
Yönlendirme çıkışı yayınlama/abone olma örneği	`amqspsra.c`	Bir yöneltme çıkışta yayınlama/abone olma iletisi için kuyruk ya da kuyruk yöneticisi hedefinin nasıl değiştirileceğini gösterir.

Java

Java örneği `MQPubSubApiSample.java` , yayınlayıcıyı ve aboneleri tek bir programda birleştirir. Kaynak ve derlenmiş sınıf dosyaları `wmqjava` örnekleri klasöründe bulunur.

If you choose to run in client mode, first see “Örnek programların hazırlanması ve çalıştırılması” sayfa 104 for details.

Konfigürasyonu tanımlanmış bir Java ortamınız varsa, Java komutunu kullanarak komut satırından örneği çalıştırın. Örneği, önceden ayarlanmış bir Java programlama çalışma ortamı olan WebSphere MQ Explorer Eclipse çalışma alanında da çalıştırabilirsiniz.

Örnek programın özelliklerini değiştirmek için bazı özelliklerin bazılarını değiştirmeniz gerekebilir. Bunu, JVM ' ye parametreler sağlayarak ya da kaynak olarak düzenleme yaparak yapabilirsiniz.

“MQPubSubApiSample Java örneğinin çalıştırılması” sayfa 130 içindeki yönergeler, örneğin Eclipse çalışma alanından nasıl çalıştırılacağını gösterir.

MQPubSubApiSample Java örneğinin çalıştırılması

How to run the MQPubSubApiSample using the Java Development Tools from the Eclipse platform.

Başlamadan önce

Eclipse çalışma ortamını açın. Yeni bir çalışma alanı dizini yaratın ve seçin. Hoş geldiniz penceresini kapatın.

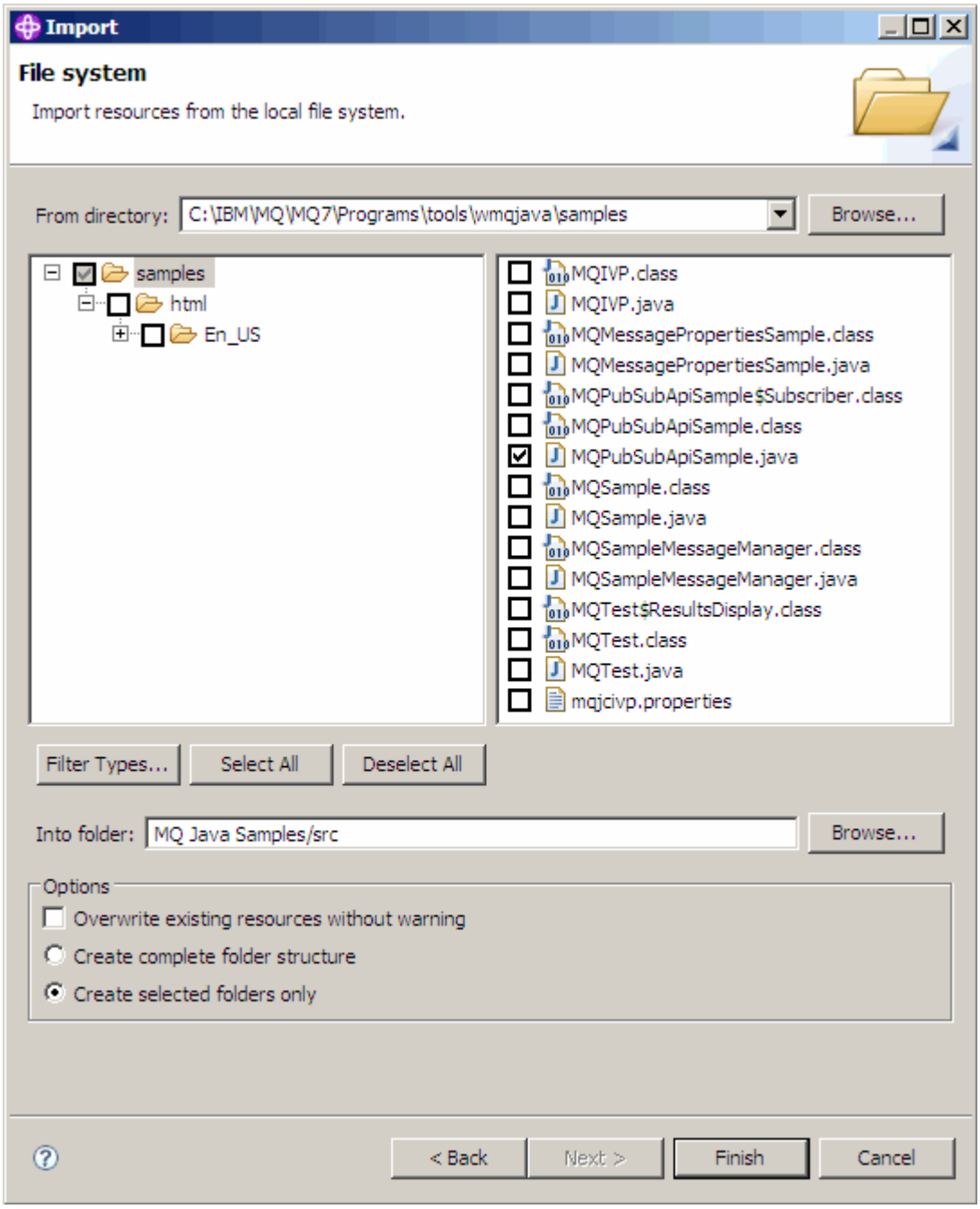
İstemci olarak çalışmadan önce “Örnek programların hazırlanması ve çalıştırılması” sayfa 104 içindeki adımları izleyin.

Bu görev hakkında

Java yayınlama/abone olma örnek programı, bir WebSphere MQ MQI istemcisi Java programıdır. Bu örnek, 1414 numaralı bağlantı noktasında bulunan varsayılan kuyruk yöneticisi kullanılarak değişiklik yapılmaksızın çalışır. Bu görev, bu basit vakayı açıklar ve genel olarak, parametrelerin nasıl sağlanacağını ve örneğinin farklı WebSphere MQ yapılandırmalarına uygun olarak nasıl değiştirileceğini belirtir. Bu örnek, Pencerelerüzerinde gösterilen şekilde gösterilmektedir. Dosya yolları diğer platformlarda farklılık gösterir.

Yordam

1. Java örnek programlarını içe aktar
   a) Çalışma ortamında, **Pencere** > **Perspektifi aç** > **Diğer** > **Java** öğelerini tıklatın ve **Tamam**düğmesini tıklatın.
   b) **Paket Gezgini** görünümüne geçin.
   c) **Package Explorer** (Paket Gezgini) görünümünde yer alan beyaz alan için sağ tıklatın. **Yeni** > **Java projesi**öğelerini tıklatın.
   d) **`Project name`** alan tipinde `MQ Java Samples`. **İleri**düğmesini tıklatın.
   e) **`Java Settings`** panelinde, **Libraries** (Kitaplıklar) sekmesine geçin.
   f) **Dış JAR Ekle**seçeneğini tıklatın.
   g) Browse to *MQ_INSTALLATION_PATH*`\java\lib` where *MQ_INSTALLATION_PATH* is the WebSphere MQ installation folder and select `com.ibm.mq.jar` and `com.ibm.mq.jmqi.jar`
   h) **Aç** > **Bitir.**öğesini tıklatın
   i) **Paket Gezgini** görünümünde `src` öğesini farenin sağ düğmesiyle tıklatın.
   j) Seç **İçe Aktar ...** > **Genel** > **Dosya Sistemi** > **Sonraki** > **Göz At**...
   ve *MQ_INSTALLATION_PATH*`\tools\wmqjava\samples` yoluna göz atın; burada *MQ_INSTALLATION_PATH* , WebSphere MQ kuruluş dizinidir.
   k) **Import** (İçe Aktarma) panosunda Şekil 20 sayfa 132, `samples` (onay kutusunu işaretlemeyin) seçeneğini tıklatın.
   l) `MQPubSubApiSample.java`seçeneğini belirleyin. **`Into folder`** alanı `MQ Java Samples/src`içermelidir. **Bitir**'i tıklatın.



Şekil 20. Dosya sistemi içe aktarma

2. Yayınlama/abone olma örnek programını çalıştırın.

   Varsayılan parametreleri değiştirmenize gerek olup olmadığına bağlı olarak, programı çalıştırmanın iki yolu vardır.

   - İlk seçenek, herhangi bir değişiklik yapmadan programı çalıştırır:
     - Çalışma alanı ana menüsünde, `src` klasörünü genişletin. **MQPubSubApiSample.javaRun-as > 1 seçeneğini sağ tıklatın. Java Uygulaması**
   - İkinci seçenek, programı değiştirgelerle ya da ortamınıza ilişkin değiştirilmiş kaynak kodla çalıştırır:
     - `MQPubSubApiSample.java`'ı açın ve `MQPubSubApiSample` oluşturucusunu çalışın.
     - Programın özniteliklerini değiştirin.

Bu öznitelikler, -D JVM anahtarı kullanılarak değiştirilebilir ya da kaynak kodu düzenleyerek Sistem özelliği için bir varsayılan değer sağlanarak değiştirilebilir.

- topicObject
- queueManagerAdı
- subscriberCount

Bu öznitelikler yalnızca oluşturucudaki kaynak kod düzenlenerek değiştirilebilir.

- hostname
- kapı
- channel

Sistem özelliklerini ayarlamak için, erişimcide varsayılan bir değer kodu yazın; örneğin:

```
queueManagerName = System.getProperty("com.ibm.mq.pubSubSample.queueManagerName",
"QM3");
```

Or provide the parameter to the JVM using the `-D` option, as shown in the following steps:

a. Ayarlamak istediğiniz System.Property ' nin tam adını kopyalayın, örneğin: `com.ibm.mq.pubSubSample.queueManagerName`.

b. Çalışma alanında, **Çalıştır** > **Çalıştır İletişim Kutularını Aç**' ı sağ tıklatın. **Uygulamaları yaratma, yönetme ve çalıştırma** ' ta Java Uygulaması 'nı çift tıklatın ve **(x) = Bağımsız Değişkenler** sekmesini tıklatın.

c. **VM bağımsız değişkenleri:** bölmesinde, `-D` yazın ve System.property adını, `com.ibm.mq.pubSubSample.queueManagerName`eve ardından =QM3adını yapıştırın. **Uygula** > **Çalıştır**seçeneğini tıklatın.

d. Virgülle ayrılmış bir liste olarak ya da bölmede ek satırlar olarak, virgül ayırıcılar olmadan ek bağımsız değişkenler ekleyin.

Örneğin:`-Dcom.ibm.mq.pubSubSample.queueManagerName=QM3,`
`-Dcom.ibm.mq.pubSubSample.subscriberCount=6`.

Publish Exit örnek programı

AMQSPSE0 , bir aboneye teslim edilmeden önce yayını kesmek için çıkılan bir çıkışa ait örnek bir C programıdır. Bundan sonra çıkış, örneğin ileti üstbilgilerini, bilgi yükünü ya da hedefi değiştirebilir ya da iletinin bir aboneye yayınlanmasını engelleyebilir.

Örneği çalıştırmak için aşağıdaki görevleri gerçekleştirin:

1. Kuyruk yöneticisini yapılandır:
   - UNIX and Linux sistemlerinde, `qm.ini` dosyasına benzer bir stanza ekleyin:

     ```
     PublishSubscribe:
               PublishExitPath=<Module>
               PublishExitFunction=EntryPoint
     ```

     Burada modül *MQ_INSTALLATION_PATH*`/samp/bin/amqspse` . *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder. Windows 'ta, eşdeğer öznitelikleri kayıt defterinde ayarlayın.
2. Modüle WebSphere MQiçin erişilebilir olduğundan emin olun.
3. Yapılandırmayı almak için Kuyruk Yöneticisini yeniden başlatın.
4. İzlenecek uygulama sürecinde, izleme dosyalarının nereye yazılacağı açıklanmalıdır. Örneğin:
   - UNIX and Linux sistemlerinde,`/var/mqm/trace` dizininin var olduğundan ve aşağıdaki ortam değişkenini dışa aktardığından emin olun:

     ```
     export MQPSE_TRACE_LOGFILE=/var/mqm/trace/PubTrace
     ```

- Windows 'ta, `C:\temp` dizininin var olduğundan ve aşağıdaki ortam değişkenini ayarlandığından emin olun:

```
set MQPSE_TRACE_LOGFILE=C:\temp\PubTrace
```

Put Sample programlar

Put örnek programları, MQPUT çağrısını kullanan bir kuyruğa ileti yerleştirdi.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Put Sample programının tasarımı

Program, iletileri koymak için hedef kuyruğu açmak için MQOO_Output seçeneğiyle MQOPER çağrısını kullanır.

Kuyruk açılamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler. Programı basit tutmak için, bu konuda ve sonraki MQI çağrılarında, program seçeneklerin çoğu için varsayılan değerleri kullanır.

Her giriş satırı için, program metni bir arabelleğe okur ve MQPUT çağrısını kullanarak, o satırın metnini içeren bir veri paketi iletisi yaratır. Program, girişin sonuna ulaşıncaya kadar ya da MQPUT çağrısının başarısız olduğu zamana kadar devam eder. Program girişin sonuna ulaşırsa, MQCLOSE çağrısını kullanarak kuyruğu kapatır.

Koyma örnek programları çalıştırılıyor

amqsput ve amqsputc örneklerinin çalıştırılması

Bu programlar her biri 2 parametre alır:

1. Hedef kuyruğun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)

Bir kuyruk yöneticisi belirtilmezse, amqsput varsayılan kuyruk yöneticisine bağlanır ve amqsputc, bir ortam değişkeniyle ya da istemci kanal tanımlama dosyası tarafından tanıtılan kuyruk yöneticisine bağlanır. Bu programları çalıştırmak için aşağıdakilerden birini girin:

- `amqsput myqueue qmanagername`
- `amqsputc myqueue qmanagername`

Burada `myqueue` , iletilerin yerleştirilecek kuyruğun adıdır; `qmanagername` ise, `myqueue`' un sahibi olan kuyruk yöneticidir.

amq0put örneğinin çalıştırılması

COBOL sürümünün herhangi bir parametresi yok. Bu, varsayılan kuyruk yöneticisine bağlanır ve çalıştırdığınız zaman sizden sorulur:

```
Please enter the name of the target queue
```

StdIn ' den giriş alır ve her giriş satırını hedef kuyruğa ekler. Boş bir satır, daha fazla veri olmadığını belirtir.

Reference Message Sample programlar

The Reference Message samples allow a large object to be transferred from one node to another (usually on different systems) without the need for the object to be stored on WebSphere MQ queues at either the source or the destination nodes.

Başvuru İletilerinin bir kuyruğa nasıl konabileceğini, ileti çıkışları tarafından alınan ve bir kuyruktan nasıl alınabileceğini göstermek için bir dizi örnek program sağlanır. Örnek programlar, dosyaları taşımak

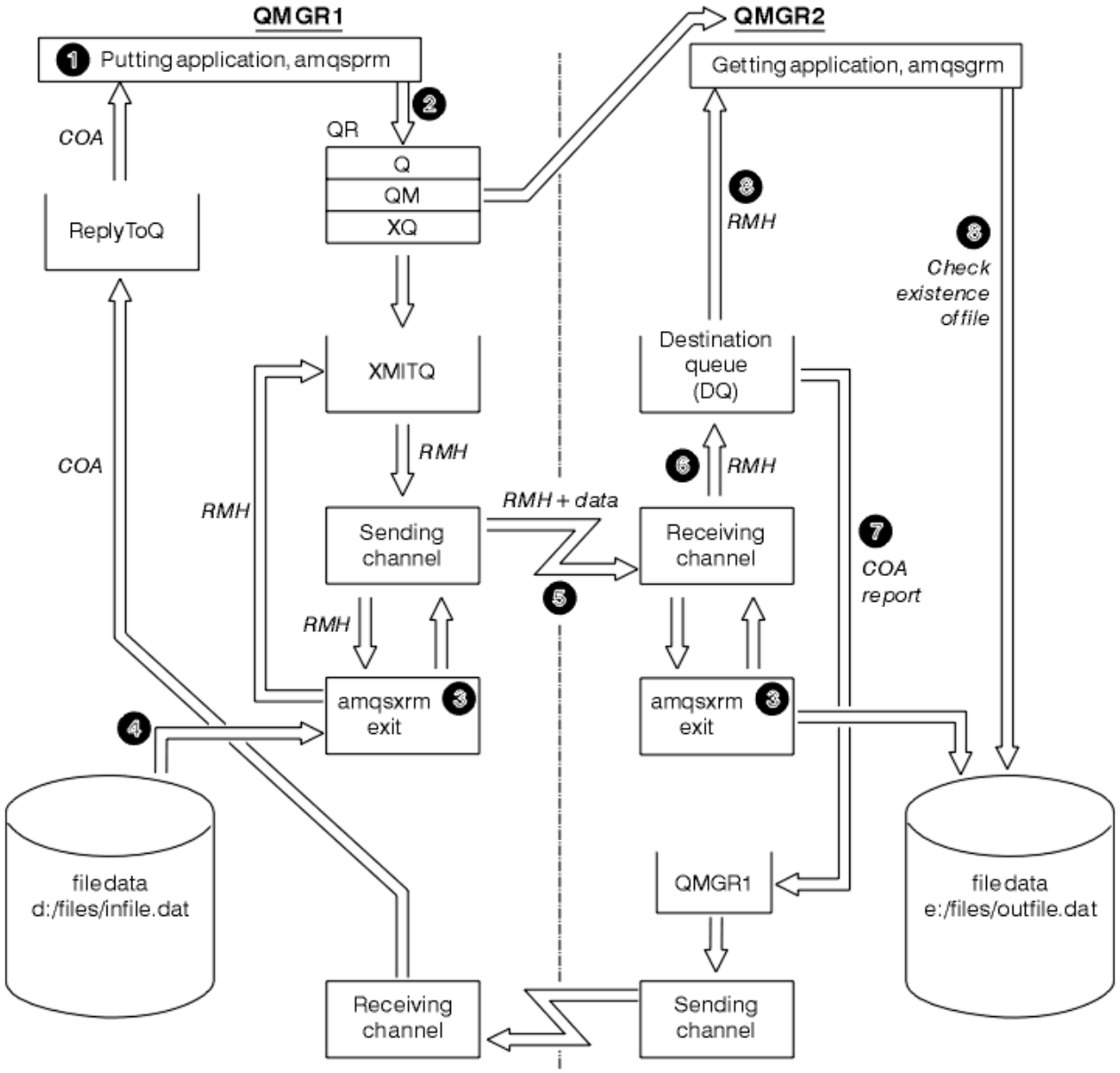
için Başvuru İletilerini kullanır. Veritabanları gibi diğer nesneleri taşımak ya da güvenlik denetimleri gerçekleştirmek istiyorsanız, örneğimize dayalı olarak kendi çıkışınızı tanımlayın amqsxrm. Aşağıdaki kısımlarda Başvuru İletisi örnek programları açıklanmıştır.

Kullanılacak Reference Message exit örnek programının sürümü, kanalın üzerinde çalışmakta olduğu altyapıya bağlıdır. Tüm altyapılarda, gönderme bitişindeki amqsxrma kullanın. Use amqsxrma at the receiving end if the receiver is running under any WebSphere MQ product except WebSphere MQ for IBM i.

Başvuru İletisi örneklerinin çalıştırılması

Başvuru İletisi örnek programlarının nasıl çalıştırılacağını öğrenmek için bu bilgileri kullanın.

Reference Message Samples komutu aşağıdaki gibi çalışır:



Şekil 21. Başvuru İletisi örneklerinin çalıştırılması

1. Dinleyicileri, kanalları ve tetikleme izleyicilerini başlatmak ve kanallarınızı ve kuyruklarınızı tanımlamak için ortamı ayarlayın.

   For the purposes of describing how to set up the Reference Message example this refers to the sending machine as MACHINE1 with a queue manager called QMGR1 and the receiving machine as MACHINE2 with a queue manager called QMGR2.

Not: The following definitions allow a Reference Message to be built to send a file with an object type of FLATFILE from queue manager QMGR1 to QMGR2 and to re-create the file as defined in the call to AMQSPRM (or AMQSPRMA on IBM i). Başvuru İletisi (dosya verileri de içinde olmak üzere), kanal CHL1 ve iletim kuyruğu XMITQ kullanılarak gönderilir ve kuyruk DQ ' ya yerleştirilir. Exception and COA reports are sent back to QMGR1 using the channel REPORT and transmission queue QMGR1.

Başvuru İletisi 'ni alan uygulama (AMQSGRM), INITQ başlatma kuyruğu ve PROC işlemi kullanılarak tetiklenir. CONNAME alanlarının doğru olarak ayarlandığından ve MSGEXIT alanının, makine tipine ve WebSphere MQ ürününün kurulu olduğu yere bağlı olarak dizin yapılarınızı yansıttığınızdan emin olun.

The MQSC definitions have used an AIX style for defining the exits. FLATFILE ileti veri FLATFILE 'ın büyük ve küçük harfe duyarlı olduğunu ve büyük harfle çalışmadığı sürece çalışmayacağını unutmayın.

Makine üzerinde MACHINE1, kuyruk yöneticisi QMGR1

MQSC sözdizimi

```
define chl(chl1) chltype(sdr) trptype(tcp) conname('machine2') xmitq(xmitq)
msgdata(FLATFILE) msgexit('/usr/lpp/mqm/samp/bin/amqsxrm(MsgExit)
')

define ql(xmitq) usage(xmitq)

define chl(report) chltype(rcvr) trptype(tcp) replace

define qr(qr) rname(dq) rqmname(qmgr2) xmitq(xmitq) replace
```

Not: Kuyruk yöneticisi adı belirtmezseniz, sistem varsayılan kuyruk yöneticisini kullanır.

```
CRTMQMCHL  CHLNAME(CHL1) CHLTYPE(*SDR) MQMNAME(QMGR1) +
             REPLACE(*YES) TRPTYPE(*TCP) +
             CONNAME('MACHINE2(60501)') TMQNAME(XMITQ) +
             MSGEXIT(QMQM/AMQSXRM4) MSGUSRDATA(FLATFILE)

CRTMQMQ    QNAME(XMITQ) QTYPE(*LCL) MQMNAME(QMGR1) +
             REPLACE(*YES) USAGE(*TMQ)

CRTMQMCHL  CHLNAME(REPORT) CHLTYPE(*RCVR) +
               MQMNAME(QMGR1) REPLACE(*YES) TRPTYPE(*TCP)

CRTMQMQ    QNAME(QR) QTYPE(*RMT) MQMNAME(QMGR1) +
             REPLACE(*YES) RMTQNAME(DQ) +
             RMTMQMNAME(QMGR2) TMQNAME(XMITQ)
```

Makine üzerinde MACHINE2, kuyruk yöneticisi QMGR2

MQSC sözdizimi

```
define chl(chl1) chltype(rcvr) trptype(tcp)
msgexit('/usr/lpp/mqm/samp/bin/amqsxrm(MsgExit)')
        msgdata(flatfile)

define chl(report) chltype(sdr) trptype(tcp) conname('MACHINE1')
        xmitq(qmgr1)

define ql(initq)

define ql(qmgr1) usage(xmitq)

define pro(proc) applicid('/usr/lpp/mqm/samp/bin/amqsgrm')

define ql(dq) initq(initq) process(proc) trigger trigtype(first)
```

2. Bir kez WebSphere MQ nesneleri oluşturulduktan sonra:
   a. Platforma uygun olduğunda, gönderen ve alıcı kuyruk yöneticilerine ilişkin dinleyiciyi başlatın
   b. Start the channels CHL1 and REPORT
   c. Alıcı kuyruk yöneticisininde, INITQ başlatma kuyruğu için tetikleme izleyicisini başlatır

3. Aşağıdaki parametreleri kullanarak, komut satırında Referans İletisi örnek programını AMQSPRM komutunu çalıştırın:

   -m Yerel kuyruk yöneticisinin adı; varsayılan kuyruk yöneticisi varsayılan değer olarak kullanılır

   -i Kaynak dosyanın adı ve yeri

   -o Hedef dosyanın adı ve yeri

   -q Kuyruğun adı

   -g -q parametresinde tanımlanan kuyruğun var olduğu kuyruk yöneticisinin adı; bu varsayılan değer olarak -m parametresinde belirtilen kuyruk yöneticisi için

   -t Nesne tipi

   -w Bekleme aralığı, yani, kural dışı durum için bekleme süresi ve alıcı kuyruk yöneticisinden COA raporları

   Örneğin, önceden tanımlanan nesnelerle birlikte örneği kullanmak için aşağıdaki parametreleri kullanırdınız:

   ```
   -mQMGR1 -iInput File -oOutput File -qQR -tFLATFILE -w120
   ```

   Bekleme süresini artırmak, büyük bir dosyanın, iletileri zamanaşımına yatırmadan önce bir ağ üzerinden gönderilmesine olanak tanır.

   ```
   amqsprm -q QR -m QMGR1 -i d:\x\file.in -o d:\y\file.out -t FLATFILE
   ```

   Not: UNIX and Linux altyapılarında, hedef dosya dizinini belirtmek için, bir yerine iki ters eğik çizgi (\\) kullanmanız gerekir. Bu nedenle, **amqsprm** komutu şu şekilde görünür:

   ```
   amqsprm -i /files/infile.dat -o e:\\files\\outfile.dat -q QR
   -m QMGR1 -w 30 -t FLATFILE
   ```

   Conference Message programının çalıştırılması aşağıdaki gibi olur:

   - Başvuru İletisi, QMGR1kuyruk yöneticisine QR kuyruğuna konmaya neden oldu.
   - Kaynak dosya ve yol `d:\files\infile.dat` ' dir ve örnek komutun verildiği sistemde bulunur.
   - QR kuyruğu uzak bir kuyruksa, Başvuru İletisi başka bir kuyruk yöneticisine, farklı bir sistemde (`e:\files\outfile.dat`adı ve yolu ile yaratılan bir dosyanın bulunduğu bir kuyruk yöneticisine gönderilir). Bu dosyanın içeriği kaynak dosyayla aynı.
   - amqsprm, hedef kuyruk yöneticisinden bir COA raporu için 30 saniye bekler.
   - Nesne tipi `flatfile`olduğundan, iletileri kuyruktan QR kuyruğundan taşımak için kullanılan kanal bunu *MsgData* alanında belirtmelidir.
4. Kanallarınızı tanımladığınızda, amqsxrm olacak şekilde hem gönderen hem de alma uçlarında ileti çıkışını seçin. This is defined on WebSphere MQ for Windows as follows:

   ```
   msgexit('pathname\amqsxrm.dll(MsgExit)')
   ```

   Bu, WebSphere MQ for AIX, WebSphere MQ for HP-UXve WebSphere MQ for Solaris için aşağıda belirtilen şekilde tanımlanır:

   ```
   msgexit('pathname/amqsxrm(MsgExit)')
   ```

   Bir yol adı belirtiyorsanız, tam adı belirtin. Yol adını atlarsanız, programın `qm.ini` dosyasında belirtilen yolda olduğu varsayılır (ya da Pencereleriçin WebSphere MQ ' ta, kayıtta belirtilen yol).
5. Kanal çıkışı, Başvuru İletisi üstbilgisini okur ve başvurduğu dosyayı bulur.

6. Daha sonra, kanal çıkışı, üstbilgiyle birlikte kanalı aşağı göndermeden önce dosyayı bölebilir. On WebSphere MQ for AIX, WebSphere MQ for HP-UX, and WebSphere MQ for Solaris, change the group owner of the target directory to 'mqm' so that the sample message exit can create the file in that directory. Ayrıca, hedef dizinin izinlerini, mqm grubu üyelerinin bu gruba yazmasına izin verecek şekilde değiştirin. Dosya verileri, WebSphere MQ kuyruklarında saklanmaz.
7. Dosyanın son bölümü alan ileti çıkışı tarafından işlendiğinde, Başvuru İletisi, amqsprm tarafından belirlenen hedef kuyruğa konmaktadır. Bu kuyruk tetiklenirse (yani, tanımlama *Trigger*, *InitQ*ve *Process* kuyruk özniteliklerini belirtiyorsa), hedef kuyruğun PROC parametresi tarafından belirlenen program tetiklenir. Tetiklenecek program, *Process* özniteliğinin *ApplId* alanında tanımlanmalıdır.
8. Başvuru İletisi hedef kuyruğa (DQ) eriştiğinde, bir COA raporu koyma uygulamasına (amqsprm) geri gönderilir.
9. Get Reference Message Sample, amqsgrm, giriş tetikleyicisi iletisinde belirtilen kuyruktan ileti alır ve kütüğün varolup olmadığını denetler.

Put Reference Message Sample tasarımı (amqsprma.c, AMQSPRM4)

Bu konu, bir Put Reference İleti örneğine ilişkin ayrıntılı bir açıklama verir.

Bu örnek, bir dosyaya gönderme yapan ve dosyayı belirtilen bir kuyruğa koyan bir Başvuru İletisi oluşturur:

1. Bu örnek, MQCONN kullanarak yerel bir kuyruk yöneticisine bağlanır.
2. Daha sonra, rapor iletilerini almak için kullanılan bir model kuyruğunu açar (MQOPER).
3. Örnek, dosyayı taşımak için gerekli olan değerleri (örneğin, kaynak ve hedef dosya adları ve nesne tipi) içeren bir Başvuru İletisi oluşturur. Örnek olarak, WebSphere MQ ile verilen örnek, `d:\x\file.in` dosyasını QMGR1 'dan QMGR2 ' a göndermek ve aşağıdaki parametreleri kullanarak dosyayı `d:\y\file.out` olarak yeniden oluşturmak için bir Başvuru İletisi oluşturur:

   ```
   amqsprm -q QR -m QMGR1 -i d:\x\file.in -o d:\y\file.out -t FLATFILE
   ```

   Burada QR , QMGR2üzerinde bir hedef kuyruğa gönderme yapan uzak bir kuyruk tanımlamasıdır.

   Not: UNIX and Linux altyapılarında, hedef dosya dizinini belirtmek için, bir yerine iki ters eğik çizgi (\\) kullanın. Bu nedenle, **amqsprm** komutu şu şekilde görünür:

   ```
   amqsprm -q QR -m QMGR1 -i /x/file.in -o d:\\y\\file.out -t FLATFILE
   ```
4. Başvuru İletisi, /q parametresi tarafından belirtilen kuyruğa (herhangi bir dosya verisi olmadan) yerleştirilir. Bu bir uzak kuyruksa, ileti ilgili iletim kuyruğuna konabilirlir.
5. Örneğin, /w parametresinde belirtilen süre (varsayılan olarak 15 saniye olarak), COA raporlarında belirtilen süre boyunca, yerel kuyruk yöneticisinde (QMGR1) yaratılan dinamik kuyruğa geri gönderilmek üzere COA raporları için bekleme süresi boyunca bekler.

Reference Message Exit örneğinin tasarımı (amqsxrma.c, AMQSXRM4)

Bu örnek, kanal tanımlamasının ileti çıkışı kullanıcı veri alanındaki nesne tipiyle eşleşen bir nesne tipine sahip Başvuru İletilerini tanır.

Bu iletiler için aşağıdakiler gerçekleşir:

- Gönderen ya da sunucu kanalındaki belirtilen veri uzunluğu, belirtilen dosyanın belirtilen göreli konumundan, Başvuru İletisi 'nden sonra aracı arabelleğindeki boşluğa kopyalanır. Dosyanın sonuna ulaşılmamışsa, Başvuru İletisi, *DataLogicalOffset* alanını güncelledikten sonra iletim kuyruğuna geri konmaktadır.
- İstekte bulunan ya da alıcı kanalısında, *DataLogicalOffset* alanı sıfırsa ve belirtilen dosya yoksa, yaratılır. Başvuru İletisi 'nin ardından gelen veriler, belirtilen dosyanın sonuna eklenir. Başvuru İletisi belirtilen dosya için son ileti değilse, atılır. Tersi durumda, hedef kuyruğa konabilmek için sonuna veri eklenmeden kanal çıkışa döndürülür.

Gönderen ve sunucu kanalları için, giriş başvuru iletisinde *DataLogicalLength* alanı sıfırsa, dosyanın geri kalan kısmı, *DataLogicalOffset* ' dan dosyanın sonuna kadar, kanal boyunca gönderilir. Bu değer sıfır değilse, yalnızca belirlenen uzunluk değeri gönderilir.

Bir hata oluşursa (örneğin, örnek bir dosyayı açamıyorsa), MQCXP.*ExitResponse* , işlenmekte olan iletinin hedef kuyruğa devam etmek yerine, ölü-mektup kuyruğuna konması için MQXCC_SUPPRESS_FUNCTION değerine ayarlıdır. MQCXP ' de bir geribildirim kodu döndürülür.*Feedback* ve iletiyi, bir rapor iletisinin ileti tanımlayıcısının *Feedback* alanına koyan uygulamaya geri döndürüldü. Bunun nedeni, MQMD ' nin *Report* alanında MQRO_EXCEPTION ayarını MQRO_EXCEPTION belirleyerek kural dışı durum raporları isteğinde bulunmasıdır.

If the encoding or *CodedCharacterSetId* (CCSID) of the Reference Message is different from that of the queue manager, the Reference Message is converted to the local encoding and CCSID. Örneğimizde, amqsprm, nesnenin biçimi MQFMT_STRING biçimidir; bu nedenle amqsxrm, veriler dosyaya yazılmadan önce nesne verilerini alan uçtaki yerel CCSID ' ye dönüştürür.

Dosya çok baytlı karakterler (örneğin, DBCS ya da Unicode) içeriyorsa, bu dosyanın MQFMT_STRING olarak aktarılmakta olduğunu belirtmeyin. Bunun nedeni, dosyanın gönderilmesi sırasında birden çok byte 'lık bir karakterin bölünebilmesinden kaynaklanır. Bu tür bir dosyayı aktarmak ve dönüştürmek için, biçimi MQFMT_STRING dışında bir değer olarak belirtin; böylece, Reference Message exit (Başvuru İletisi çıkışı) işlevi dönüştürmez ve aktarma tamamlandığında dosyayı alıcı uçta dönüştürmesini sağlar.

Reference Message Exit örneğinin derlenmesi

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

amqsxrma ' yı derlemek için aşağıdaki komutları kullanın:

Açık AIX

```
xlc_r -q64 -e MsgExit -bE:amqsxrm.exp -bM:SRE -o amqsxrm_64_r
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -lmqm_r amqsqrma.c
```

HP-UXüzerinde

```
$ c89 +DD64 +z -c -D_HPUX_SOURCE -o amqsxrma.o amqsqrma.c -IMQ_INSTALLATION_PATH/inc
$ ld -b amqsxrma.o -o /var/mqm/exits64/amqsxrma -LMQ_INSTALLATION_PATH/lib64
-L/usr/lib/pa20_64 -lmqm_r -lpthread
```

AçıkLinux

```
$ gcc -m64 -shared -fPIC -o /var/mqm/exits64/amqsxrma amqsqrma.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -lmqm_r
```

Solaris üzerinde

```
$ cc -xarch=v9 -mt -G -o /var/mqm/exits64/amqsxrma amqsqrma.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -lmqm

 -lsocket
-lnsl -ldl
```

Pencerelerüzerinde

WebSphere MQ artık mqm kitaplığını istemci paketleriyle birlikte sunucu paketleriyle birlikte sağlar; bu nedenle, aşağıdaki örnek `mqmvx.lib`yerine `mqm.lib` kullanır:

```
cl amqsqrma.c /link /out:amqsxrm.dll /dll mqm.lib mqm.lib /def:amqsxrm.def
```

Kanal çıkışlarını yazma ve derlemeye ilişkin genel bilgiler için bkz. “Kanal-çıkış programları yazılıyor” sayfa 381

Get Reference Message Sample (amqsgrma.c, AMQSGRM4) tasarımı

Bu konuda, Get Reference Message örneğinin tasarımı açıklanmaktadır.

Program mantığı aşağıdaki gibidir:

1. Örnek tetiklenir ve giriş tetikleyicisi iletisinden kuyruk ve kuyruk yöneticisi adlarını alır.
2. Daha sonra MQCONN kullanarak belirlenen kuyruk yöneticisine bağlanır ve MQOPED kullanarak belirtilen kuyruğu açar.
3. Örnek sorunlar, kuyruktan ileti almak için bir döngü içinde 15 saniye bekleme aralığıyla MQGET ' i yayınlar.
4. İleti bir Başvuru İletisi ise, örnek aktarılmış olan dosyanın varlığını denetler.
5. Daha sonra kuyruk kapatılır ve kuyruk yöneticisinden bağlantıyı keser.

İstek örnek programları

İstek örnek programları, istemci/sunucu işlemesini gösterir. Örnekler, bir sunucu programı tarafından işlenen bir hedef sunucu kuyruğuna istek iletileri yerleştiren istemcilerdir. Sunucu programının bir yanıtlama kuyruğuna yanıt iletisi göndermesini bekler.

İstek örnekleri, MQPUT çağrısını kullanarak hedef sunucu kuyruğuna bir dizi istek iletisi yerleştirdi. Bu iletiler yerel kuyruğu belirtir, SYSTEM.SAMPLE.REPLY , yerel ya da uzak bir kuyruk olabilen yanıt kuyruğu olarak. Programlar yanıt iletilerini bekler, sonra bunları görüntüler. Yanıtlar yalnızca, hedef sunucu kuyruğu bir sunucu uygulaması tarafından işleniyorsa ya da bu amaçla bir uygulama tetiklendiyse gönderilir (Sorgula, Ayarla, Echo örnek programları tetiklenecek şekilde tasarlanmıştır). C örneği 1 dakika bekler (COBOL örneği 5 dakika bekler), ilk yanıt için gelen ilk yanıt (bir sunucu uygulamasının tetiklenmesine izin vermek için) ve sonraki yanıtlar için 15 saniye bekler, ancak her iki örnek de yanıt almadan sona erebilir. İstek örnek programlarının adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

İstek örnek programlarının çalıştırılması

amqsreq0.c, amqsreq ve amqsreqc örnekleri çalıştırılıyor

Programın C sürümü üç parametre alır:

1. Hedef sunucu kuyruğunun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)
3. Yanıt kuyruğu (isteğe bağlı)

Örneğin, aşağıdakilerden birini girin:

- `amqsreq myqueue qmanagername replyqueue`
- `amqsreqc myqueue qmanagername`
- `amq0req0 myqueue`

Burada `myqueue` hedef sunucu kuyruğunun adı, `qmanagername` , `myqueue`' un sahibi olan kuyruk yöneticisinin adıdır; `replyqueue` ise yanıt kuyruğunun adıdır.

Kuyruk yöneticisinin adını atarsanız, kuyruğun varsayılan kuyruk yöneticisinin iyesi olduğu varsayılır. Yanıt kuyruğunun adını atlarsanız, varsayılan yanıt kuyruğu sağlanır.

amq0req0.cbl örneğinin çalıştırılması

COBOL sürümünün herhangi bir parametresi yok. Bu, varsayılan kuyruk yöneticisine bağlanır ve çalıştırdığınız zaman sizden sorulur:

```
Please enter the name of the target server queue
```

Program, girişini StdIn öğesinden alır ve her bir satırı, bir istek iletisinin içeriği olarak her metin satırını alarak hedef sunucu kuyruğuna ekler. Boş bir satır okunduğunda program sona erer.

AMQSREQ4 örneğinin çalıştırılması

C programı, boş bir zaman sonlandırma girişi içeren stdin ' den (klavye) veri alarak iletiler oluşturur. Program üç değiştirge alır: Hedef kuyruğun adı (gerekli), kuyruk yöneticisi adı (isteğe bağlı) ve yanıtlama kuyruğu adı (isteğe bağlı). Kuyruk yöneticisi adı belirlenmezse, varsayılan kuyruk yöneticisi kullanılır. Herhangi bir yanıt kuyruğu belirlenmezse, SYSTEM.SAMPLE.REPLY kuyruğu kullanıldı.

Burada, yanıt kuyruğunu belirten, ancak kuyruk yöneticisi varsayılan değerinin belirlenmesine olanak vermek için C örnek programının nasıl çağrılacağı bir örneği yer alıyor:

```
CALL PGM(QMQM/AMQSREQ4) PARM('SYSTEM.SAMPLE.LOCAL' '' 'SYSTEM.SAMPLE.REPLY')
```

Not: Kuyruk adlarının büyük ve küçük harfe duyarlı olduğunu unutmayın. Örnek dosya yaratma programı AMQSAMP4 tarafından yaratılan tüm kuyruklar, büyük harfli karakterlerde yaratılmış adlara sahiptir.

AMQ0REQ4 örneğinin çalıştırılması

COBOL programı, klavyeden verileri kabul ederek iletiler oluşturur. Programı başlatmak için, programı çağırın ve parametre olarak hedef kuyruğunuzun adını belirtin. Program klavyeden bir arabelleğe giriş kabul eder ve her metin satırı için bir istek iletisi oluşturur. Klavye üzerine boş bir satır girdiğinizde program durur.

Tetikleme kullanılarak İstek örneği çalıştırılıyor

Örnek, tetikleme, Sorgu, Ayar ya da Yankı örnek programlarından biri ile kullanılırsa, giriş satırı, tetiklenen programın erişmesini istediğiniz kuyruğun kuyruk adı olmalıdır.

UNIX, Linux ve Windows sistemleri

Örnekleri tetiklemeyi kullanarak çalıştırmak için:

1. Bir oturumda RUNMQTRM tetikleme izleme programını başlatır (başlatma kuyruğu SYSTEM.SAMPLE.TRIGGER kullanılabilir.)
2. amqsreq programını başka bir oturumda başlatın.
3. Hedef sunucu kuyruğu tanımladığınızdan emin olun.

   İletileri yerleştirmek için istek örneği için hedef sunucu kuyruğu olarak kullanmak üzere kullanabileceğiniz örnek kuyruklar şunlardır:

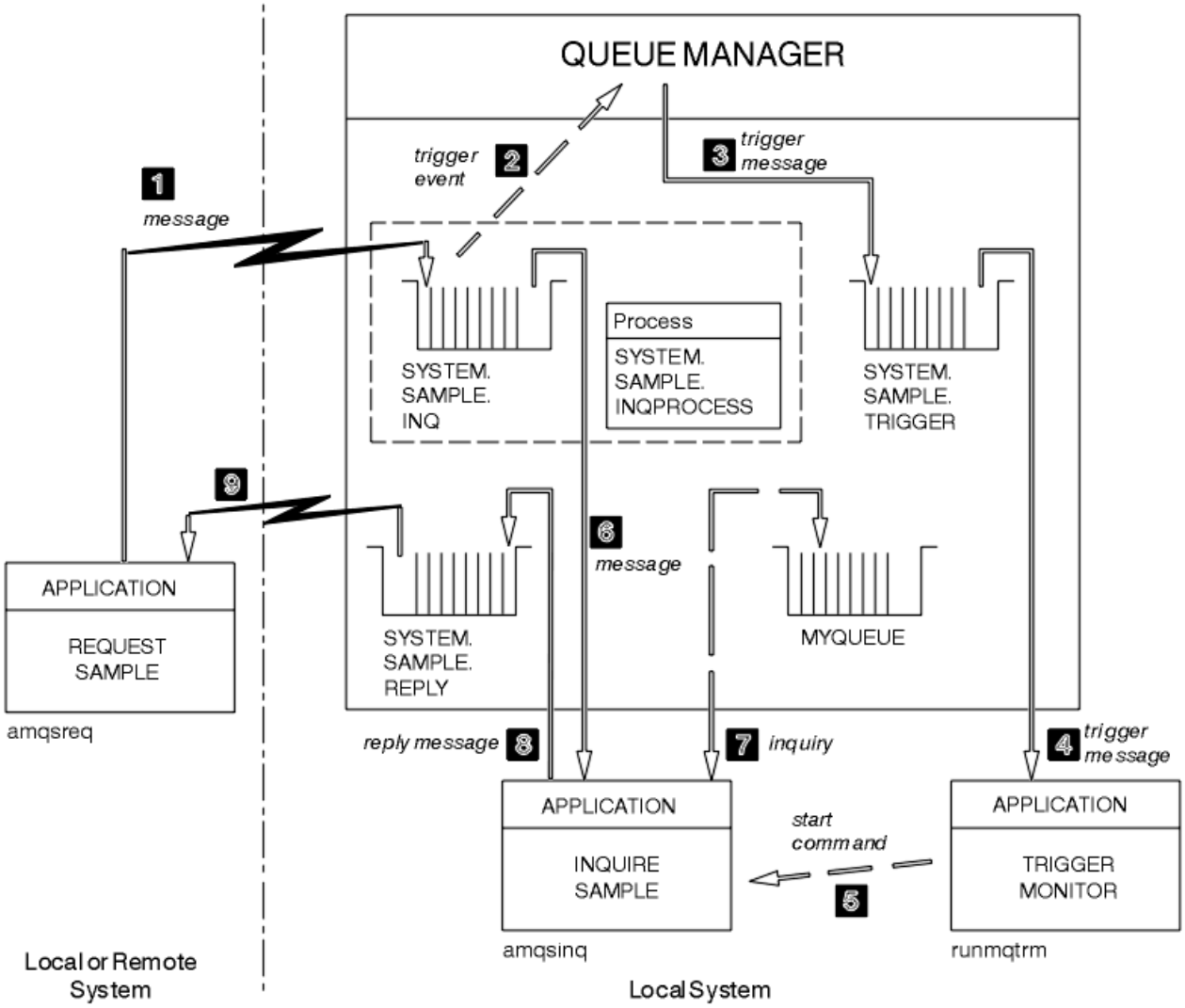
   - SYSTEM.SAMPLE.INQ -Sorgula ile ilgili örnek program için
   - SYSTEM.SAMPLE.SET -Küme örnek programı için
   - SYSTEM.SAMPLE.ECHO -Yankı örnek programı için

   Bu kuyruklar FIRST tetikleyicisi tipine sahiptir; bu nedenle, İstek örneğini çalıştırmadan önce kuyruklarda önceden iletiler varsa, sunucu uygulamaları gönderdiğiniz iletiler tarafından tetiklenmez.
4. Kullanmak için, Sorgula, Set ya da Echo örnek programı için bir kuyruk tanımladığınızdan emin olun.

Bu, istek örneği bir ileti gönderdiğinde tetikleme izleyicinin hazır olduğu anlamına gelir.

Not: RUNMQSC ve amqscos0.tst dosyası kullanılarak yaratılan örnek süreç tanımlamaları C örneklerini tetikler. amqscos0.tst içindeki süreç tanımlamalarını değiştirin ve COBOL sürümlerini kullanmak için bu güncellenmiş dosya ile RUNMQSC komutunu kullanın.

Şekil 22 sayfa 142 , İsteme ve Sorgunun örneklerini birlikte nasıl kullanacağını gösterir.



Şekil 22. Tetikleme kullanarak istek ve sorgulama örnekleri

Şekil 22 sayfa 142 ' da İstek örneği, iletileri hedef sunucu kuyruğuna (SYSTEM.SAMPLE.INQve sorgulamak için kuyruğun, MYQUEUE. Alternatively, you can use one of the sample queues defined when you ran amqscos0.tst, or any other queue that you have defined, for the Inquire sample.

Not: Şekil 22 sayfa 142 içindeki sayılar olayların sırasını gösterir.

İsteği ve Sorgula sorgulama örneklerini çalıştırmak için şu tetiklemeyi kullanın:

1. Kullanmak istediğiniz kuyrukların tanımlandığından emin olun. Örnek kuyrukları tanımlamak için amqscos0.tstkomutunu çalıştırın ve bir kuyruk MYQUEUE öğesini tanımlayın.
2. RUNMQTRM tetikleme izleyicisini çalıştırın:

   ```
   RUNMQTRM -m qmanagerıname -q SYSTEM.SAMPLE.TRIGGER
   ```

3. İstek örneğini çalıştır

   ```
   amqsıreq SYSTEM.SAMPLE.INQ
   ```

   Not: Tetiklenecek olanları süreç nesnesi tanımlar. İstemci ve sunucu aynı altyapıda çalışmıyorsa, tetikleme izleyicisinin başlattığı tüm işlemler *ApplType*tanımlamalıdır; tersi durumda, sunucu varsayılan tanımlarını (yani, normalde sunucu makinesiyle ilişkili uygulama tipi) alır ve bir hataya neden olur.

Uygulama tiplerinin listesi için bkz. ApplType.

4. Sorgulamaya ilişkin örneğinin kullanmasını istediğiniz kuyruğun adını girin:

```
MYQUEUE
```

5. Boş bir satır girin (İstek programını sona erdirmek için).
6. Bundan sonra, istek örneği, MYQUEUE ' dan elde edilen sorgulamak programı verilerini içeren bir ileti görüntüler.

Birden çok kuyruk kullanabilirsiniz; bu durumda, “4” sayfa 143adımındaki diğer kuyrukların adlarını girin.

Tetikleme ile ilgili daha fazla bilgi için bkz. “Starting IBM WebSphere MQ applications using triggers” sayfa 313.

İstek örnek programının tasarımı

Program, iletileri koyabilmesi için hedef sunucu kuyruğunu açar. MQOO_OUTPUT seçeneğiyle MQOPEN çağrısını kullanır. Kuyruğu açamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler.

Daha sonra program, SYSTEM.SAMPLE.REPLY , böylece yanıt iletileri alabilirler. Bunun için, program MQOO_INPUT_EXCLUSIVE seçeneği ile MQOPEN çağrısını kullanır. Kuyruğu açamazsa, program, MQOPED çağrısının döndürdüğü neden kodunu içeren bir hata iletisi görüntüler.

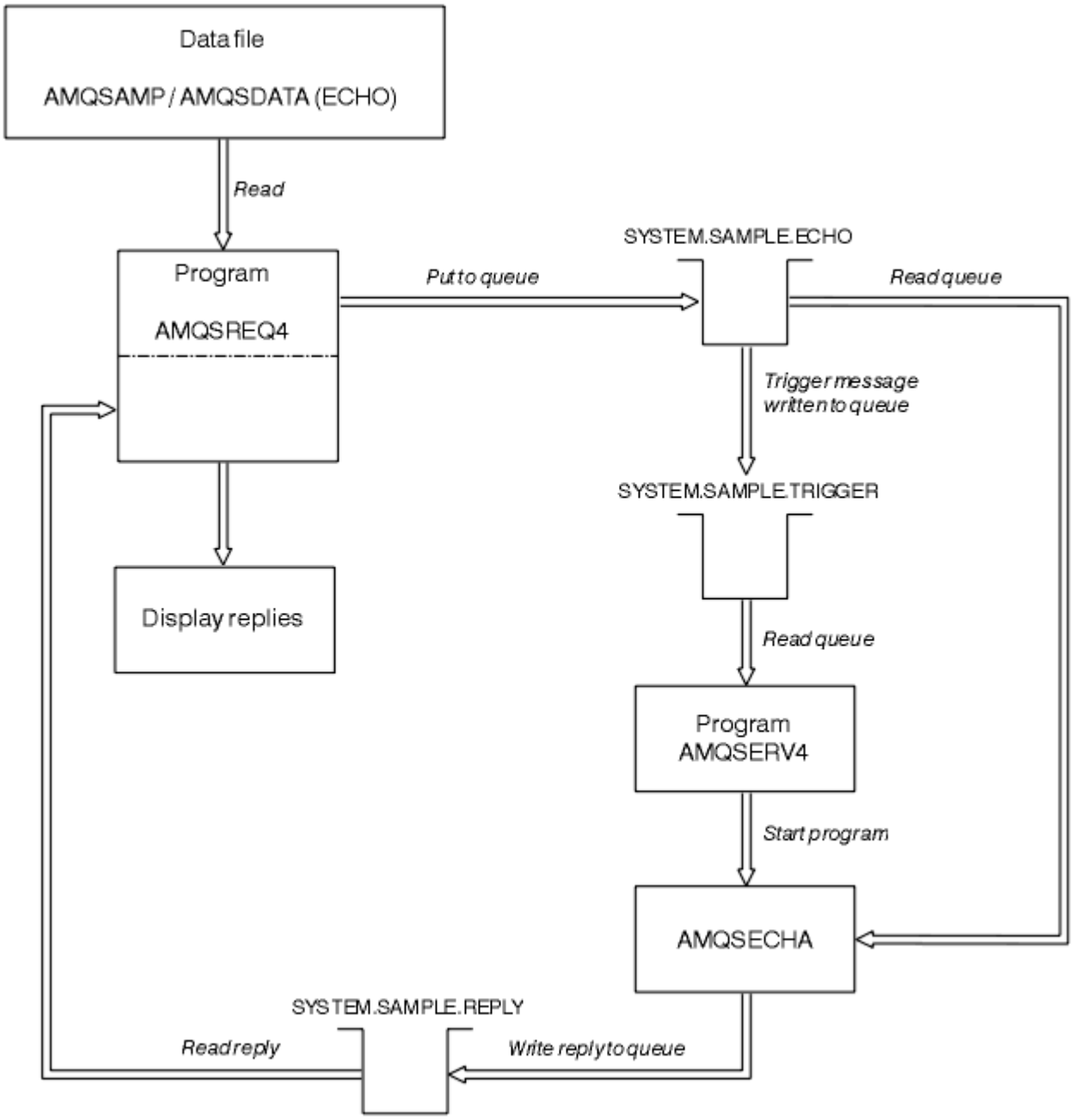
Her giriş satırı için, program metni bir arabelleğe okur ve MQPUT çağrısını kullanarak, o satırın metnini içeren bir istek iletisi yaratır. Bu çağrıda program, istek iletisine ilişkin gönderilen herhangi bir rapor iletisinin ileti verilerinin ilk 100 baytı içermesini istemek için MQRO_EXCEPTION_WITH_DATA rapor seçeneğini kullanır. Program, girişin sonuna ulaşıncaya kadar ya da MQPUT çağrısının başarısız olduğu zamana kadar devam eder.

Daha sonra, program yanıtlama iletilerini kuyruktan kaldırmak için MQGET çağrısını kullanır ve yanıtlarda yer alan verileri görüntüler. MQGET çağrısı MQGMO_WAIT, MQGMO_CONVERT ve MQGMO_ACCEPT_KISALTILMIŞ seçenekleri kullanır. *WaitInterval* , COBOL sürümünde 5 dakika ve C sürümünde 1 dakika, ilk yanıt için (bir sunucu uygulamasının tetiklenmesine izin vermek için) ve sonraki yanıtlar için 15 saniye. Bu süre, kuyruğun üzerinde herhangi bir ileti yoksa, bu dönemleri bekler. Bu aralığın süresi dolmadan bir ileti gelmezse, çağrı başarısız olur ve MQRC_NO_MSG_AVAILEABLE neden kodunu döndürür. Çağrı ayrıca, bildirilen arabellek büyüklüğünden daha uzun iletilerin kısaltılması için MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini de kullanır.

Bu program, bu alanları, aldığı iletide yer alan değerlere ayarlandığından, her MQGET çağrısından sonra MQMD yapısının *MsgId* ve *CorrelId* alanlarının nasıl temizleneceğini gösterir. Bu alanların temizlenmesi, art arda gelen MQGET çağrılarının iletilerin kuyrukta tutulmakta olduğu sırayla alma çağrıları anlamına gelir.

Bu program, MQGET çağrısına ilişkin MQRC_NO_MSG_AVAILEABLE neden kodunu döndürünceye ya da MQGET çağrısının başarısız olduğu zamana kadar devam eder. Arama başarısız olursa, program neden kodunu içeren bir hata iletisi görüntüler.

Daha sonra, program, MQCLOSE çağrısını kullanarak hem hedef sunucu kuyruğunu, hem de yanıtlamayı kuyruğa kapatır.



Şekil 23. Örnek IBM i Client/Server (Echo) programı akış şeması

Örnek programları ayarla

Küme örnek programları, kuyruğun *InhibitPut* özniteliğini değiştirmek için MQSET çağrısını kullanarak işlemleri bir kuyruğa yerleştirmeyi engeller. Ayrıca, Set Sample programlarının tasarımıyla ilgili bilgi edinin.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Programların tetiklenen programlar olarak çalıştırılması amaçlanır, bu nedenle bunların tek girişi, sorgulanacak özniteliklere sahip bir hedef kuyruğun adını içeren bir MQTMC2 (tetikleme iletisi) yapısıdır. C sürümü kuyruk yöneticisi adını da kullanır. COBOL sürümü, varsayılan kuyruk yöneticisini kullanır.

Tetikleme işleminin çalışması için, kullanmak istediğiniz Set Sample programının, SYSTEM.SAMPLE.SETkuyruğuna gelen iletiler tarafından tetiklendiğinden emin olun. To do this, specify the name of the Set sample program that you want to use in the *ApplicId* field of the process definition SYSTEM.SAMPLE.SETPROCESS. Örnek kuyruğun bir FIRST tetikleyicisi tipi vardır; İstek örneğini

çalıştırmadan önce kuyruğunda önceden iletiler varsa, Küme örneği gönderdiğiniz iletiler tarafından tetiklenmez.

Tanımlamayı doğru bir şekilde ayarladığınızda:

- UNIX, Linux ve Windows sistemleri için, bir oturumda **runmqtrm** programını başlatın ve amqsreq programını başka bir oturum içinde başlatın.
- IBM iiçin, bir oturumda AMQSERV4 programını başlatın ve ardından başka bir oturum içinde AMQSREQ4 programını başlatın. AMQSERV4yerine AMQSTRG4 ' yi kullanabilirsiniz, ancak olası iş gönderimi gecikmeleri, gerçekleşenleri takip etmeyi daha az kolaylaştırabilirdi.

İstek örnek programlarını kullanarak, her biri yalnızca bir kuyruk adı içeren istek iletilerini SYSTEM.SAMPLE.SETkuyruğuna yollamak için kullanın. Her istek iletisi için, Set Sample Programs (Örnek Programları Ayarla), belirtilen kuyruk üzerinde işlem engellenmiş olan bir doğrulama içeren bir yanıt iletisi gönderir. Yanıtlar, istek iletisinde belirtilen yanıtlama kuyruğuna gönderilir.

Set Sample programının tasarımı

Program, tetikleme iletisi yapısında adı belirtilen kuyruğu açarken, bu işlem başlatıldığında iletileceği bir kuyrukta yer alan kuyruğu açar. (For clarity, we will call this the *istek kuyruğu*.) Program, bu kuyruğu paylaşılan giriş için açmak için MQOPEN çağrısını kullanır.

Program, bu kuyruktan iletileri kaldırmak için MQGET çağrısını kullanır. Bu çağrı, 5 saniye bekleme süresi ile MQGMO_ACCEPT_TRUNCATED_MSG ve MQGMO_WADET seçeneklerini kullanır. Program, bir istek iletisi olup olmadığını görmek için her iletinin tanımlayıcısını sınar; değilse, program iletiyi atar ve bir uyarı iletisi görüntüler.

İstek kuyruğundan kaldırılan her istek iletisi için, program, verilerde bulunan kuyruğun adını (*hedef kuyruk*adını çağıracak) okur ve MQOO_SET seçeneğiyle MQOPER çağrısını kullanarak bu kuyruğu açar. Program daha sonra, hedef kuyruğun *InhibitPut* özniteliğinin değerini MQQA_PUT_INHIBITED olarak ayarlamak için MQSET çağrısını kullanır.

MQSET çağrısı başarılı olursa, program yanıt kuyruğuna yanıt iletisi koymak için MQPUT1 çağrısını kullanır. Bu ileti, `PUT inhibited`ddizgisini içerir.

MQAUT ya da MQSET çağrısı başarısız olursa, program yanıt kuyruğuna `report` iletisi koymak için MQPUT1 çağrısını kullanır. Bu rapor iletisinin ileti tanımlayıcısının *Feedback* alanında, başarısız olan buna bağlı olarak, MQOPEN ya da MQSET çağrısının döndürdüğü neden kodudur.

MQSET çağrısından sonra program, MQCLOSE çağrısını kullanarak hedef kuyruğu kapar.

İstek kuyruğunda bir ileti kalmadığında, program o kuyruğu kapatır ve kuyruk yöneticisinden bağlantıyı keser.

SSL/TLS örnek programı

AMQSSLC, MQCONNX çağrısında SSL/TLS istemci bağlantısı bilgilerini sağlamak için MQCNO ve MQSCO yapılarının nasıl kullanılacağını gösteren örnek bir C programıdır. Bu, istemci bir MQI uygulamasının istemci bağlantı kanalı tanımını ve SSL/TLS ayarlarını bir istemci kanal tanımlama çizelgesi (CCDT) olmadan çalıştırma zamanında sağlamasını sağlar.

Bir bağlantı adı belirtilirse, program bir MQCD yapısında istemci bağlantısı kanalı tanımlaması oluşturur.

Anahtar havuzu dosyasının kök adı belirtilirse, program bir MQSCO yapısı oluşturur; OCSP yanıtlayıcı URL 'si de sağlanırsa, program bir kimlik doğrulama bilgisi kaydı MQAIR yapısını oluşturur.

Daha sonra, program MQCONNX komutunu kullanarak kuyruk yöneticisine bağlanır. Bağlantı kuruldu ve bağlı olduğu kuyruk yöneticisinin adını yazdırır.

Bu programın bir MQI istemci uygulaması olarak bağlanması amaçlanır. Ancak, olağan bir MQI uygulaması olarak bağlantılandırılabilir. Daha sonra, yerel bir kuyruk yöneticisine bağlanır ve istemci bağlantısı bilgilerini yoksayar.

AMQSSLC aşağıdaki değiştirgeleri kabul eder; bunların tümü isteğe bağlıdır:

-m QmgrName
Bağlanılacak kuyruk yöneticisinin adı

-c ChannelName
Kullanılacak kanalın adı

-x ConnName
Sunucu bağlantısı adı

SSL/TLS parametreleri:

-k KeyReposStem
Anahtar havuzu dosyasının kök adı. Bu, .kdb soneki olmadan dosyanın tam yoludur. Örneğin:

```
/home/user/client
C:\User\client
```

-s CipherSpec
Kuyruk yöneticindeki SVRCONN kanal tanımlamasındaki SSLCIPH ' ye karşılık gelen SSL/TLS kanalı CipherSpec dizgisi.

-f
Yalnızca FIPS 140-2 sertifikalı algoritmaların kullanılması gerektiğini belirtir.

-b VALUE1[,VALUE2...]
Yalnızca Suite B uyumlu algoritmaların kullanılması gerektiğini belirtir. Bu değiştirge, şu değerlerden birinin ya da birkaçının virgülle ayrılmış bir listesidir: NONE,128_BIT,192_BIT. Bu değerler, MQSUITEB ortam değişkeni ile aynı anlamı ve istemci yapılandırma dosyasındaki eşdeğer EncryptionPolicySuiteB ayarına sahip olur.

-p İlkesi
Kullanılacak sertifika doğrulama ilkesini belirtir. Bu, aşağıdaki değerlerden biri olabilir:

HERHANGİ BİRİ
Güvenli yuva kitaplığı tarafından desteklenen sertifika geçerlilik denetimi ilkelerinin her birini uygulayın ve herhangi bir ilkenin sertifika zincirini geçerli olarak kabul etmesi durumunda sertifika zincirini kabul edin. Bu ayar, modern sertifika standartlarıyla uyumlu olmayan eski dijital sertifikalar ile geriye dönük uyumluluk üst sınırı için kullanılabilir.

RFC5280
Yalnızca RFC 5280 uyumlu sertifika geçerlilik denetimi ilkesini uygulayın. Bu ayar ANY ayarından daha katı geçerlilik denetimi sağlar, ancak bazı eski dijital sertifikaları reddeder.

Varsayılan değer ANY değeridir.

OCSP sertifikası iptal parametresi:

-o URL
OCSP Yanıtlayıcı URL 'si

SSL/TLS örnek programının çalıştırılması
SSL/TLS örnek programını çalıştırmak için önce SSL ya da TLS ortamınızı ayarlamanız gerekir. Daha sonra, bir dizi parametre belirterek, örneği komut satırından çalıştırıyorsunuz.

Bu görev hakkında
Aşağıdaki yönergeler, örnek programı kişisel sertifikalar kullanarak çalıştırır. Komut, örneğin, CA sertifikalarını kullanabilir ve bir OCSP yanıtlayıcısı kullanarak durumlarının denetlenmesini sağlar. Örnek içindeki yönergelere bakın.

Yordam
1. QM1adını taşıyan bir kuyruk yöneticisi yaratın. Daha fazla bilgi için bkz. crtmqm.
2. Kuyruk yöneticisi için bir anahtar havuzu yaratın. Daha fazla bilgi için bkz. UNIX, Linux, and Windows sistemlerinde anahtar havuzu ayarlama.

3. İstemci için bir anahtar havuzu yaratın. Call it *clientkey.kdb*.
4. Kuyruk yöneticisi için kişisel bir sertifika yaratın. Daha fazla bilgi için UNIX, Linux, and Windows sistemlerinde kendinden onaylı kişisel sertifika oluşturmabaşlıklı konuya bakın.
5. İstemci için kişisel bir sertifika yaratın.
6. Sunucu anahtar havuzundan kişisel sertifikayı alın ve istemci havuzuna ekleyin. Daha fazla bilgi için bkz. UNIX, Linux ve Pencereler sistemlerinde anahtar havuzundan, kendinden onaylı sertifikana ilişkin genel kısmının çekilmesi ve UNIX, Linux ya da Pencereler sistemlerinde CA sertifikası (ya da kendinden onaylı bir sertifikenin genel bölümü) bir anahtar havuzuna eklenmesi.
7. İstemci anahtar havuzundan kişisel sertifikayı alın ve sunucu anahtar havuzuna ekleyin.
8. MQSC komutunu kullanarak bir sunucu bağlantı kanalı yaratın:

   ```
   DEFINE CHANNEL(QM1SVRCONN) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCIPH(NULL_SHA)
   ```

   Ek bilgi için bkz. Sunucu bağlantı kanalı
9. Kuyruk yöneticinde bir kanal dinleyicisi tanımlayın ve başlatın. Ek bilgi için DEFINE LISTENER ve START LISTENERbaşlıklı konuya bakın.
10. Aşağıdaki komutu kullanarak örnek programı çalıştırın:

    ```
    AMQSSSLC -m QM1 -c QM1SVRCONN -x localhost
    -k "c:\Program Files\IBM\WebSphere MQ\clientkey" -s NULL_SHA
    -o http://dummy.OCSP.responder
    ```

Sonuçlar

Örnek program aşağıdaki işlemleri gerçekleştirir:

1. Belirtilen herhangi bir kuyruk yöneticisine ya da varsayılan kuyruk yöneticisine, belirtilen seçenekleri kullanarak bağlanır.
2. Kuyruk yöneticisini açar ve adını sorgular.
3. Kuyruk yöneticisini kapatır.
4. Kuyruk yöneticisinden bağlantıyı keser.

Örnek program başarıyla çalıştırılırsa, çıkışı aşağıdaki örneğe benzer şekilde görüntüler:

```
Sample AMQSSSLC start
Connecting to queue manager QM1
Using the server connection channel QM1SVRCONN
on connection name localhost.
Using SSL CipherSpec NULL_SHA
Using SSL key repository stem c:\Program Files\IBM\WebSphere MQ\clientkey
Using OCSP responder URL http://dummy.OCSP.responder
Connection established to queue manager QM1

Sample AMQSSSLC end
```

Örnek program bir sorunla karşılaşırsa, uygun bir hata iletisi görüntüler; örneğin, geçersiz bir OCSP yanıtlayıcısı URL 'si belirtirseniz, aşağıdaki iletiyi alırsınız:

```
MQCONNX ended with reason code 2553
```

Neden kodlarının bir listesi için bkz. API neden kodları.

Tetikleme örnek programları

Tetikleme örneğinde sağlanan işlev, **runmqtrm** programındaki tetikleyici izleyicide sağlanan bir altkümedir.

Bu programların adları için bkz. “Örnek programlar içinde gösterilen özellikler” sayfa 93 .

Tetikleme örneğinin tasarımı

Tetikleme örneği programı, MQOO_INPUT_AS_Q_DEF seçeneğiyle MQOPEN çağrısını kullanarak başlatma kuyruğunu açar. Sınırsız bekleme aralığı belirterek, MQGET çağrısını MQGMO_accept_truncated_msg ve MQGMO_WAWT seçenekleriyle kullanarak, başlatma kuyruğundan iletiler alır. Program, iletileri sırayla almak için her MQGET çağrısından önce *MsgId* ve *CorrelId* alanlarını temizler.

Başlatma kuyruğundan bir ileti alındığında, program iletinin büyüklüğünü denetleyerek, bu iletinin bir MQTM yapısıyla aynı boyutta olduğundan emin olmak için iletiyi sınar. Bu sınama başarısız olursa, program bir uyarı görüntüler.

Geçerli tetikleyici iletiler için, tetikleme örneği şu alanlardaki verileri kopyalar: *ApplicId*, *EnvrData*, *Version*, ve *ApplType*. Bu alanların son ikisi sayısaldır, bu nedenle program, UNIX, Linuxve Windows sistemleri için MQTMC2 yapısında kullanılacak karakter değiştirmeleri yaratır.

Tetikleme örneği, tetikleme iletisinin *ApplicId* alanında belirtilen uygulamaya bir başlangıç komutu verir ve bir MQTMC2 ya da MQTMC (tetikleme iletisinin karakter sürümü) yapısını geçirir. In UNIX, Linux and Pencereler systems, the *EnvData* field is used as an extension to the invoking command string.

Son olarak, program başlatma kuyruğunu kapatır.

Tetikleme örnek programlarının çalıştırılması

Bu konu, tetikleme örneği programlarının çalıştırılmasına ilişkin bilgiler içerir.

amqstrg0.c, amqstrg ve amqstrgc örneklerinin çalıştırılması

Program 2 parametre alır:

1. Başlatma kuyruğunun adı (gerekli)
2. Kuyruk yöneticisinin adı (isteğe bağlı)

Kuyruk yöneticisi belirtilmediyse, varsayılan değer olarak varsayılan değer olarak bağlanır. A sample initiation queue will have been defined when you ran amqscos0.tst; the name of that queue is SYSTEM.SAMPLE.TRIGGER, and you can use it when you run this program.

Not: Bu örnekteki işlev, **runmqtrm** programında sağlanan tam tetikleme işlevinin bir alt kümesidir.

Tetikleme sunucusunun tasarımı

Tetikleme sunucusunun tasarımı tetikleme izleyicisine benzer; tetikleme sunucusu dışında:

- MQAT_CICS ' in yanı sıra MQAT_OS400 uygulamalarını sağlar
- For CICS applications, substitutes the *EnvData*, for example, to specify the CICS region, from the trigger message in the STRCICSUSR command
- Paylaşılan giriş için başlatma kuyruğunu açar; böylece, birçok tetikleme sunucusu aynı anda çalışabilir.

Not: AMQSERV4 tarafından başlatılan programlar, tetikleme sunucusunu durdurduğu için MQDISC çağrısını kullanmamalıdır. AMQSERV4 tarafından başlatılan programlar MQCONN çağrısını kullanarak başlatıldıysa, MQRC_ALREADY_CONNECTED neden kodunu alır.

SMOKIN örnekleri

Smokin için Put ve Get örnek programları hakkında bilgi edinin ve sunucu ortamını SMOKIN ' de oluşturma.

Bu örnekleri çalıştırmadan önce, sunucu ortamını oluşturmanız gerekir.

Not: Bu konu boyunca ters eğik çizgi (\) karakteri, uzun komutları birden çok satıra bölmek için kullanılır. Bu karakteri girmeyin. Her komutu tek bir satır olarak girin.

Sunucu ortamı oluşturuluyor

Farklı platformlar için WebSphere MQ için sunucu ortamını oluşturmaya ilişkin bilgiler.

Çalışan bir TUXEDO ortamınız olduğu varsayılır.

WebSphere MQ for AIX (32 bit) için sunucu ortamı oluşturulması

1. Sunucu ortamının oluşturulduğu bir dizin (örneğin, < APPDIR>) oluşturun ve bu dizindeki tüm komutları yürütün.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , SMOKIN için kök dizin ve *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder:

```
$ export CFLAGS="-I MQ_INSTALLATION_PATH□inc -I /<APPDIR> -L MQ_INSTALLATION_PATH□lib"
$ export LDOPTS="-lmqm"
$ export FIELDTBLS=MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ export VIEWFILES=/<APPDIR>/amqstxvx.V
$ export LIBPATH=$TUXDIR□lib:MQ_INSTALLATION_PATH□lib:□lib
```

3. Aşağıdaki bilgileri TUXEDO dosyasına udataobj/RM dosyasına ekleyin

```
MQSeries_XA_RMI:MQRMIXASwitchDynamic: -lmqmxa -lmqm
```

4. Şu komutları çalıştırın:

```
$ mkfldhdr    MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ viewc       MQ_INSTALLATION_PATH/samp/amqstxvx.v
$ buildtms    -o MQXA -r MQSeries_XA_RMI
$ buildserver -o MQSERV1 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.a \
       -r MQSeries_XA_RMI -s MPUT1:MPUT \
       -s MGET1:MGET \
       -v -bshm
$ buildserver -o MQSERV2 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.a \
       -r MQSeries_XA_RMI -s MPUT2:MPUT
       -s MGET2:MGET \
       -v -bshm
$ buildclient -o doputs  -f MQ_INSTALLATION_PATH/samp/amqstxpx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.a
$ buildclient -o dogets  -f MQ_INSTALLATION_PATH/samp/amqstxgx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.a
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
$ tmloadcf    -y MQ_INSTALLATION_PATH/samp/ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
$tmadmin -c
```

   Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
> crdl -z /<APPDIR>/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
$ strmqm
```

8. Smokin Başlat:

```
$ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

WebSphere MQ for AIX (64 bit) için sunucu ortamı oluşturulması

1. Sunucu ortamının oluşturulduğu bir dizin (örneğin, < APPDIR>) oluşturun ve bu dizindeki tüm komutları yürütün.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , SMOKIN için kök dizini temsil eder ve *MQ_INSTALLATION_PATH* , WebSphere MQ ' un installed.:

```
       $ export CFLAGS="-I MQ_INSTALLATION_PATH□inc -I /<APPDIR> -L
MQ_INSTALLATION_PATH□lib64"
       $ export LDOPTS="-lmqm"
       $ export FIELDTBLS=MQ_INSTALLATION_PATH/samp/amqstxvx.flds
       $ export VIEWFILES=/<APPDIR>/amqstxvx.V
       $ export LIBPATH=$TUXDIR□lib64:MQ_INSTALLATION_PATH□lib64:□lib64
```

3. Aşağıdaki bilgileri TUXEDO dosyasına udataobj/RM dosyasına ekleyin

```
       MQSeries_XA_RMI:MQRMIXASwitchDynamic: -lmqmxa64 -lmqm
```

4. Şu komutları çalıştırın:

```
       $ mkfldhdr    MQ_INSTALLATION_PATH/samp/amqstxvx.flds
       $ viewc       MQ_INSTALLATION_PATH/samp/amqstxvx.v
       $ buildtms    -o MQXA -r MQSeries_XA_RMI
       $ buildserver -o MQSERV1 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
              -f MQ_INSTALLATION_PATH/lib64/libmqm.a \
              -r MQSeries_XA_RMI -s MPUT1:MPUT \
              -s MGET1:MGET \
              -v -bshm
       $ buildserver -o MQSERV2 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
              -f MQ_INSTALLATION_PATH/lib64/libmqm.a \
              -r MQSeries_XA_RMI -s MPUT2:MPUT
              -s MGET2:MGET \
              -v -bshm
       $ buildclient -o doputs  -f MQ_INSTALLATION_PATH/samp/amqstxpx.c \
              -f MQ_INSTALLATION_PATH/lib64/libmqm.a
       $ buildclient -o dogets  -f MQ_INSTALLATION_PATH/samp/amqstxgx.c \
              -f MQ_INSTALLATION_PATH/lib64/libmqm.a
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
       $ tmloadcf    -y MQ_INSTALLATION_PATH/samp/ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
      $tmadmin -c
```

   Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
      > crdl -z /<APPDIR>/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
       $ strmqm
```

8. Smokin Başlat:

```
       $ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

Building the server environment for WebSphere MQ for Solaris (32-bit)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

1. Sunucu ortamının yerleşik olduğu bir dizin (örneğin, *APPDIR*) oluşturun ve bu dizindeki tüm komutları yürütür.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , TUXEDO için kök dizinidir:

```
$ export CFLAGS="-I /APPDIR"
$ export FIELDTBLS=amqstxvx.flds
$ export VIEWFILES=amqstxvx.V
$ export SHLIB_PATH=$TUXDIR/lib:MQ_INSTALLATION_PATH/lib:/lib
$ export LD_LIBRARY_PATH=$TUXDIR/lib:MQ_INSTALLATION_PATH/lib:/lib
```

3. Aşağıdaki bilgileri TUXEDO dosyası udataobj/RM ' ye ekleyin.

```
MQSeries_XA_RMI:MQRMIXASwitchDynamic: \
MQ_INSTALLATION_PATH/lib/libmqmxa.a MQ_INSTALLATION_PATH/lib/libmqm.so \
/opt/tuxedo/lib/libtux.a
```

4. Şu komutları çalıştırın:

```
$ mkfldhdr    amqstxvx.flds
$ viewc       amqstxvx.v
$ buildtms    -o MQXA -r MQSeries_XA_RMI
$ buildserver -o MQSERV1 -f amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT1:MPUT \
       -s MGET1:MGET \
       -v -bshm
       -l -ldl
$ buildserver -o MQSERV2 -f amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT2:MPUT \
       -s MGET2:MGET \
       -v -bshm
       -l -ldl
$ buildclient -o doputs  -f amqstxpx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so  \

$ buildclient -o dogets  -f amqstxgx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
$ tmloadcf    -y ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
$tmadmin -c
```

   Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
> crdl -z /APPDIR/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
$ strmqm
```

8. Smokin Başlat:

```
$ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

Building the server environment for WebSphere MQ for Solaris (64-bit)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

1. Sunucu ortamının oluşturulduğu bir dizin (örneğin, < APPDIR>) oluşturun ve bu dizindeki tüm komutları yürütün.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , TUXEDO için kök dizinidir:

```
$ export CFLAGS="-I /<APPDIR>"
$ export FIELDTBLS=amqstxvx.flds
$ export VIEWFILES=amqstxvx.V
$ export SHLIB_PATH=$TUXDIR□lib:MQ_INSTALLATION_PATH□lib:□lib64
$ export LD_LIBRARY_PATH=$TUXDIR□lib64:MQ_INSTALLATION_PATH□lib64:□lib64
```

3. Aşağıdaki bilgileri TUXEDO dosyası udataobj/RM ' ye ekleyin.

```
MQSeries_XA_RMI:MQRMIXASwitchDynamic: \
MQ_INSTALLATION_PATH/lib64/libmqmxa64.a MQ_INSTALLATION_PATH/lib64/libmqm.so \
/opt/tuxedo/lib64/libtux.a
```

4. Şu komutları çalıştırın:

```
$ mkfldhdr    amqstxvx.flds
$ viewc       amqstxvx.v
$ buildtms    -o MQXA -r MQSeries_XA_RMI
$ buildserver -o MQSERV1 -f amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT1:MPUT \
       -s MGET1:MGET \
       -v -bshm
       -l -ldl
$ buildserver -o MQSERV2 -f amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT2:MPUT \
       -s MGET2:MGET \
       -v -bshm
       -l -ldl
$ buildclient -o doputs  -f amqstxpx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so  \

$ buildclient -o dogets  -f amqstxgx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
$ tmloadcf    -y ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
$tmadmin -c
```

   Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
> crdl -z /<APPDIR>/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
$ strmqm
```

8. Smokin Başlat:

```
$ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

Building the server environment for WebSphere MQ for HP-UX (32-bit)

Not: 32 bitlik SMOKIN sunucu ortamı yalnızca Itanium platformunda oluşturulabilir.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

1. Sunucu ortamının oluşturulduğu bir dizin (örneğin, < APPDIR>) oluşturun ve bu dizindeki tüm komutları yürütün.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , TUXEDO için kök dizinidir:

```
$ export CFLAGS="-Aa -D_HPUX_SOURCE"
$ export FIELDTBLS=MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ export VIEWFILES=$APPDIR/amqstxvx.V
$ export TUXCONFIG=$APPDIR/tuxconfig
$ export PATH=$TUXDIR/bin:/usr/bin:/sbin:MQ_INSTALLATION_PATH/bin:$PATH
$ export SHLIB_PATH=$TUXDIR/lib:MQ_INSTALLATION_PATH/lib:/lib
$ export FLDTBLDIR=$APPDIR:$TUXDIR/udataobj
```

3. Aşağıdaki bilgileri TUXEDO dosyasına udataobj/RM dosyasına ekleyin

```
MQSeries_XA_RMI:MQRMIXASwitchDynamic: \
MQ_INSTALLATION_PATH/lib/libmqmxa.so MQ_INSTALLATION_PATH/lib/libmqm.so \
/opt/tuxedo/lib/libtux.sl
```

4. Şu komutları çalıştırın:

```
$ mkfldhdr    MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ viewc       MQ_INSTALLATION_PATH/samp/amqstxvx.v
```

   mkfldhdr ve viewc komutlarını çalıştırdıktan sonra, amqstxvx.h üstbilgi dosyası, SMOKIN uygulama dizininde yaratılır. Bu dosyayı TUXEDO uygulama dizininden TUXEDO include dizinine kopyalayın ve sonra aşağıdaki komutları çalıştırın.

```
$ buildtms    -o MQXA -r MQSeries_XA_RMI
$ buildserver -o MQSERV1 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT1:MPUT \
       -s MGET1:MGET \
       -v -bshm
$ buildserver -o MQSERV2 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT2:MPUT \
       -s MGET2:MGET \
       -v -bshm
$ buildclient -o doputs  -f MQ_INSTALLATION_PATH/samp/amqstxpx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so
$ buildclient -o dogets  -f MQ_INSTALLATION_PATH/samp/amqstxgx.c \
       -f MQ_INSTALLATION_PATH/lib/libmqm.so
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
$ tmloadcf    -y MQ_INSTALLATION_PATH/samp/ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
$tmadmin -c
```

   Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
> crdl -z /<APPDIR>/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
$ strmqm
```

8. Başlangıç SMOKIN:

```
$ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

Building the server environment for WebSphere MQ for HP-UX (64-bit)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

1. Sunucu ortamının oluşturulduğu bir dizin (örneğin, < APPDIR>) oluşturun ve bu dizindeki tüm komutları yürütün.
2. Aşağıdaki ortam değişkenlerini dışa aktarın; burada TUXDIR , TUXEDO için kök dizinidir:

```
$ export CFLAGS="-Aa -D_HPUX_SOURCE"
$ export FIELDTBLS=MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ export VIEWFILES=$APPDIR/amqstxvx.V
$ export TUXCONFIG=$APPDIR/tuxconfig
$ export PATH=$TUXDIR/bin:/usr/bin:/sbin:MQ_INSTALLATION_PATH/bin:$PATH
$ export SHLIB_PATH=$TUXDIR/lib:MQ_INSTALLATION_PATH/lib64:/lib64
$ export FLDTBLDIR=$APPDIR:$TUXDIR/udataobj
```

3. Aşağıdaki bilgileri TUXEDO dosyasına udataobj/RM dosyasına ekleyin

   HP-UX IA64 (IPF) platformunda:

```
MQSeries_XA_RMI:MQRMIXASwitchDynamic: \
MQ_INSTALLATION_PATH/lib64/libmqmxa64.so MQ_INSTALLATION_PATH/lib64/libmqm.so \
/opt/tuxedo/lib/libtux.sl
```

   Not: HP-UX IA64 (IPF) platformunda gönderilen WebSphere MQ kitaplıklarının bir .so dosya adı uzantısı vardır.

4. Şu komutları çalıştırın:

```
$ mkfldhdr    MQ_INSTALLATION_PATH/samp/amqstxvx.flds
$ viewc       MQ_INSTALLATION_PATH/samp/amqstxvx.v
```

   mkfldhdr ve viewc komutlarını çalıştırdıktan sonra, amqstxvx.h üstbilgi dosyası, SMOKIN uygulama dizininde yaratılır. Bu dosyayı TUXEDO uygulama dizininden TUXEDO include dizinine kopyalayın ve sonra aşağıdaki komutları çalıştırın.

```
$ buildtms    -o MQXA -r MQSeries_XA_RMI
```

   HP-UX IA64 (IPF) platformunda:

```
$ buildserver -o MQSERV1 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT1:MPUT \
       -s MGET1:MGET \
       -v -bshm
$ buildserver -o MQSERV2 -f MQ_INSTALLATION_PATH/samp/amqstxsx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so \
       -r MQSeries_XA_RMI -s MPUT2:MPUT \
       -s MGET2:MGET \
       -v -bshm
$ buildclient -o doputs  -f MQ_INSTALLATION_PATH/samp/amqstxpx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so
```

```
$ buildclient -o dogets  -f MQ_INSTALLATION_PATH/samp/amqstxgx.c \
       -f MQ_INSTALLATION_PATH/lib64/libmqm.so
```

5. ubbstxcx.cfg dosyasını düzenleyin ve gereken şekilde makine adının, çalışma dizinlerinin ve kuyruk yöneticisine ilişkin ayrıntıları ekleyin:

```
$ tmloadcf    -y MQ_INSTALLATION_PATH/samp/ubbstxcx.cfg
```

6. TLOGDEVICE ' yi yaratın:

```
$tmadmin -c
```

Bir bilgi istemi görüntülenir. Bu komut isteminde şunu girin:

```
> crdl -z /<APPDIR>/TLOG1
```

7. Kuyruk yöneticisini başlat:

```
$ strmqm
```

8. Başlangıç SMOKIN:

```
$ tmboot -y
```

Şimdi, iletileri bir kuyruğa koymak ve kuyruktan almak için, doputs ve dogets programlarını kullanabilirsiniz.

Building the server environment for WebSphere MQ for Pencereler (32-bit)

Not: <> ile tanıtılan alanları aşağıdaki dizin yollarına çevirin:

< MQMDIR>	WebSphere MQ kurulu olduğunda belirtilen dizin yolu; örneğin, `g:\Program Files\IBM\WebSphere MQ`
< TUXDIR>	SMOKIN kurulu olduğunda belirtilen dizin yolu; örneğin, `f:\tuxedo`
< APPDIR>	örnek uygulama için kullanılacak dizin yolu; örneğin, `f:\tuxedo\apps\mqapp`

Sunucu ortamı ve örnekleri oluşturmak için:

1. Örnek uygulamanın oluşturulacağı bir uygulama dizini yaratın; örneğin:

```
f:\tuxedo\apps\mqapp
```

2. Aşağıdaki örnek dosyaları WebSphere MQ örnek dizininden uygulama dizinine kopyalayın:

```
amqstxmn.mak
amqstxen.env
ubbstxcn.cfg
```

3. Bu dosyaların her birini, kuruluşunuzda kullanılan dizin adlarını ve dizin yollarını ayarlamak için düzenleyin.
4. Bağlanmak istediğiniz makine adına ve kuyruk yöneticisine ilişkin ayrıntıları eklemek için ubbstxcn.cfg dosyasını (bkz. Şekil 24 sayfa 156) düzenleyin.
5. Aşağıdaki satırı SMOKIN kütüğünün < TUXDIR > udataobj\rm dizinine ekle

```
MQSeries_XA_RMI;MQRMIXASwitchDynamic;
   <MQMDIR>\tools\lib\mqmxa.lib <MQMDIR>\tools\lib\mqm.lib
```

Burada < MQMDIR >, önceki örnekte gösterildiği gibi yerine konur. Burada iki satır olarak gösterilse de, yeni girişin dosyada bir satır olması gerekir.

6. Aşağıdaki ortam değişkenlerini ayarlayın:

```
TUXDIR=<TUXDIR>
TUXCONFIG=<APPDIR>\tuxconfig
FIELDTBLS=<MQMDIR>\tools\c\samples\amqstxvx.fld
LANG=C
```

7. SMOKIN için bir TLOG aygıtı oluşturun. Bunu yapmak için `tmadmin -c`komutunu çağırın ve komutu girin:

```
crdl -z <APPDIR>\TLOG
```

Burada <*APPDIR*> değiştirilir.

8. Geçerli dizini < APPDIR> olarak ayarlayın ve örnek makefile (amqstxmn.mak) dosyasını dış proje makefile olarak çağırın. Örneğin, Microsoft Visual C++ ile şu komutu verin:

```
msvc amqstxmn.mak
```

Tüm örnek programları oluşturmak için **oluştur** seçeneğini belirleyin.

```
*RESOURCES
IPCKEY        99999
UID           0
GID           0
MAXACCESSERS  20
MAXSERVERS    20
MAXSERVICES   50
MASTER        SITE1
MODEL         SHM
LDBAL         N

*MACHINES
<MachineName> LMID=SITE1
              TUXDIR="f:\tuxedo"
              APPDIR="f:\tuxedo\apps\mqapp;g:\Program Files\IBM\WebSphere MQ\bin"
              ENVFILE="f:\tuxedo\apps\mqapp\amqstxen.env"
              TUXCONFIG="f:\tuxedo\apps\mqapp\tuxconfig"
              ULOGPFX="f:\tuxedo\apps\mqapp\ULOG"
              TLOGDEVICE="f:\tuxedo\apps\mqapp\TLOG"
              TLOGNAME=TLOG
              TYPE="i386NT"
              UID=0
              GID=0

*GROUPS
GROUP1
        LMID=SITE1  GRPNO=1
        TMSNAME=MQXA
        OPENINFO="MQSeries_XA_RMI:MYQUEUEMANAGER"

*SERVERS
DEFAULT: CLOPT="-A -- -m MYQUEUEMANAGER"

MQSERV1     SRVGRP=GROUP1 SRVID=1
MQSERV2     SRVGRP=GROUP1 SRVID=2

*SERVICES
MPUT1
MGET1
MPUT2
MGET2
```

Şekil 24. Windowsiçin WebSphere MQ için ubbstxcn.cfg dosyası örneği

Not: Dizin adlarını ve dizin yollarını kuruluşunuzla eşleşecek şekilde değiştirin. Kuyruk yöneticisi adı MYQUEUEMANAGER adını, bağlanmak istediğiniz kuyruk yöneticisinin adına da değiştirin. Eklemeniz gereken diğer bilgiler <> karakterleriyle tanımlanır.

The sample ubbconfig file for WebSphere MQ for Pencereler is listed in Şekil 24 sayfa 156. Bu, WebSphere MQ Samples dizininde ubbstxcn.cfg olarak sağlanır.

The sample makefile (see Şekil 25 sayfa 157) supplied for WebSphere MQ for Pencereler is called ubbstxmn.mak, and is held in the WebSphere MQ samples directory.

```
TUXDIR  = f:\tuxedo
MQMDIR  = g:\Program Files\IBM\WebSphere MQ
APPDIR  = f:\tuxedo\apps\mqapp
MQMLIB  = $(MQMDIR)\tools\lib
MQMINC  = $(MQMDIR)\tools\c\include
MQMSAMP = $(MQMDIR)\tools\c\samples
INC = -f "-I$(MQMINC) -I$(APPDIR)"
DBG = -f "/Zi"

amqstx.exe:
 $(TUXDIR)\bin\mkfldhdr    -d$(APPDIR) $(MQMSAMP)\amqstxvx.fld
 $(TUXDIR)\bin\viewc       -d$(APPDIR) $(MQMSAMP)\amqstxvx.v
 $(TUXDIR)\bin\buildtms    -o MQXA -r MQSeries_XA_RMI
 $(TUXDIR)\bin\buildserver -o MQSERV1 -f $(MQMSAMP)\amqstxsx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG) \
                           -r MQSeries_XA_RMI \
                           -s MPUT1:MPUT -s MGET1:MGET
 $(TUXDIR)\bin\buildserver -o MQSERV2 -f $(MQMSAMP)\amqstxsx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG) \
                           -r MQSeries_XA_RMI \
                           -s MPUT2:MPUT -s MGET2:MGET
 $(TUXDIR)\bin\buildclient -o doputs -f $(MQMSAMP)\amqstxpx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG)
 $(TUXDIR)\bin\buildclient -o dogets -f $(MQMSAMP)\amqstxgx.c \
                           -f $(MQMLIB)\mqm.lib $(INC) -v $(DBG)
 $(TUXDIR)\bin\tmloadcf    -y $(APPDIR)\ubbstxcn.cfg
```

Şekil 25. Sample TUXEDO makefile for WebSphere MQ for Pencereler

WebSphere MQ for Windows (64 bit) için sunucu ortamı oluşturulması

Not: <> ile tanıtılan alanları aşağıdaki dizin yollarına çevirin:

< MQMDIR>	WebSphere MQ kurulu olduğunda belirtilen dizin yolu; örneğin, `g:\Program Files\IBM\WebSphere MQ`
< TUXDIR>	SMOKIN kurulu olduğunda belirtilen dizin yolu; örneğin, `f:\tuxedo`
< APPDIR>	örnek uygulama için kullanılacak dizin yolu; örneğin, `f:\tuxedo\apps\mqapp`

Sunucu ortamı ve örnekleri oluşturmak için:

1. Örnek uygulamanın oluşturulacağı bir uygulama dizini yaratın; örneğin:

   ```
   f:\tuxedo\apps\mqapp
   ```

2. Aşağıdaki örnek dosyaları WebSphere MQ örnek dizininden uygulama dizinine kopyalayın:

   ```
   amqstxmn.mak
   amqstxen.env
   ubbstxcn.cfg
   ```

3. Bu dosyaların her birini, kuruluşunuzda kullanılan dizin adlarını ve dizin yollarını ayarlamak için düzenleyin.
4. Bağlantı kurmak istediğiniz makine adının ve kuyruk yöneticisinin ayrıntılarını eklemek için `ubbstxcn.cfg` (bkz. Şekil 26 sayfa 159) başlıklı konuya bakın.

5. Add the following line to the TUXEDO file <TUXDIR>udataobj\rm

```
MQSeries_XA_RMI;MQRMIXASwitchDynamic;
   <MQMDIR>\tools\lib64\mqmxa64.lib <MQMDIR>\tools\lib64\mqm.lib
```

   Burada < MQMDIR > değiştirilir. Burada iki satır olarak gösterilse de, yeni girişin dosyada bir satır olması gerekir.

6. Aşağıdaki ortam değişkenlerini ayarlayın:

```
TUXDIR=<TUXDIR>
TUXCONFIG=<APPDIR>\tuxconfig
FIELDTBLS=<MQMDIR>\tools\c\samples\amqstxvx.fld
LANG=C
```

7. SMOKIN için bir TLOG aygıtı oluşturun. Bunu yapmak için tmadmin -ckomutunu çağırın ve komutu girin:

```
crdl -z <APPDIR>\TLOG
```

   Burada <*APPDIR*> , önceki örnekte gösterildiği gibi değiştirilir.

8. Geçerli dizini < APPDIR> olarak ayarlayın ve örnek makefile (amqstxmn.mak) dosyasını dış proje makefile olarak çağırın. Örneğin, Microsoft Visual C++ ile şu komutu verin:

```
msvc amqstxmn.mak
```

   Tüm örnek programları oluşturmak için **oluştur** seçeneğini belirleyin.

```
*RESOURCES
IPCKEY        99999
UID           0
GID           0
MAXACCESSERS  20
MAXSERVERS    20
MAXSERVICES   50
MASTER        SITE1
MODEL         SHM
LDBAL         N

*MACHINES
<MachineName> LMID=SITE1
              TUXDIR="f:\tuxedo"
              APPDIR="f:\tuxedo\apps\mqapp;g:\Program Files\IBM\WebSphere MQ\bin"
              ENVFILE="f:\tuxedo\apps\mqapp\amqstxen.env"
              TUXCONFIG="f:\tuxedo\apps\mqapp\tuxconfig"
              ULOGPFX="f:\tuxedo\apps\mqapp\ULOG"
              TLOGDEVICE="f:\tuxedo\apps\mqapp\TLOG"
              TLOGNAME=TLOG
              TYPE="i386NT"
              UID=0
              GID=0

*GROUPS
GROUP1
        LMID=SITE1  GRPNO=1
        TMSNAME=MQXA
        OPENINFO="MQSeries_XA_RMI:MYQUEUEMANAGER"

*SERVERS
DEFAULT: CLOPT="-A -- -m MYQUEUEMANAGER"

MQSERV1     SRVGRP=GROUP1 SRVID=1
MQSERV2     SRVGRP=GROUP1 SRVID=2

*SERVICES
MPUT1
MGET1
MPUT2
MGET2
```

Şekil 26. Windowsiçin WebSphere MQ için ubbstxcn.cfg dosyası örneği

Not: Dizin adlarını ve dizin yollarını kuruluşunuzla eşleşecek şekilde değiştirin. Kuyruk yöneticisi adı MYQUEUEMANAGER adını, bağlanmak istediğiniz kuyruk yöneticisinin adına da değiştirin. Eklemeniz gereken diğer bilgiler <> karakterleriyle tanımlanır.

The sample ubbconfig file for WebSphere MQ for Pencereler is listed in Şekil 26 sayfa 159. Bu, WebSphere MQ Samples dizininde ubbstxcn.cfg olarak sağlanır.

The sample makefile (see Şekil 27 sayfa 160) supplied for WebSphere MQ for Pencereler is called ubbstxmn.mak, and is held in the WebSphere MQ samples directory.

```
TUXDIR  = f:\tuxedo
MQMDIR  = g:\Program Files\IBM\WebSphere MQ
APPDIR  = f:\tuxedo\apps\mqapp
MQMLIB  = $(MQMDIR)\tools\lib64
MQMINC  = $(MQMDIR)\tools\c\include
MQMSAMP = $(MQMDIR)\tools\c\samples
INC = -f "-I$(MQMINC) -I$(APPDIR)"
DBG = -f "/Zi"

amqstx.exe:
 $(TUXDIR)\bin\mkfldhdr    -d$(APPDIR) $(MQMSAMP)\amqstxvx.fld
 $(TUXDIR)\bin\viewc       -d$(APPDIR) $(MQMSAMP)\amqstxvx.v
 $(TUXDIR)\bin\buildtms    -o MQXA -r MQSeries_XA_RMI
 $(TUXDIR)\bin\buildserver -o MQSERV1 -f $(MQMSAMP)\amqstxsx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG) \
                           -r MQSeries_XA_RMI \
                           -s MPUT1:MPUT -s MGET1:MGET
 $(TUXDIR)\bin\buildserver -o MQSERV2 -f $(MQMSAMP)\amqstxsx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG) \
                           -r MQSeries_XA_RMI \
                           -s MPUT2:MPUT -s MGET2:MGET
 $(TUXDIR)\bin\buildclient -o doputs -f $(MQMSAMP)\amqstxpx.c \
                           -f $(MQMLIB)\mqm.lib -v $(INC) $(DBG)
 $(TUXDIR)\bin\buildclient -o dogets -f $(MQMSAMP)\amqstxgx.c \
                           -f $(MQMLIB)\mqm.lib $(INC) -v $(DBG)
 $(TUXDIR)\bin\tmloadcf    -y $(APPDIR)\ubbstxcn.cfg
```

Şekil 27. Sample TUXEDO makefile for WebSphere MQ for Pencereler

SMOKIN için örnek sunucu programı

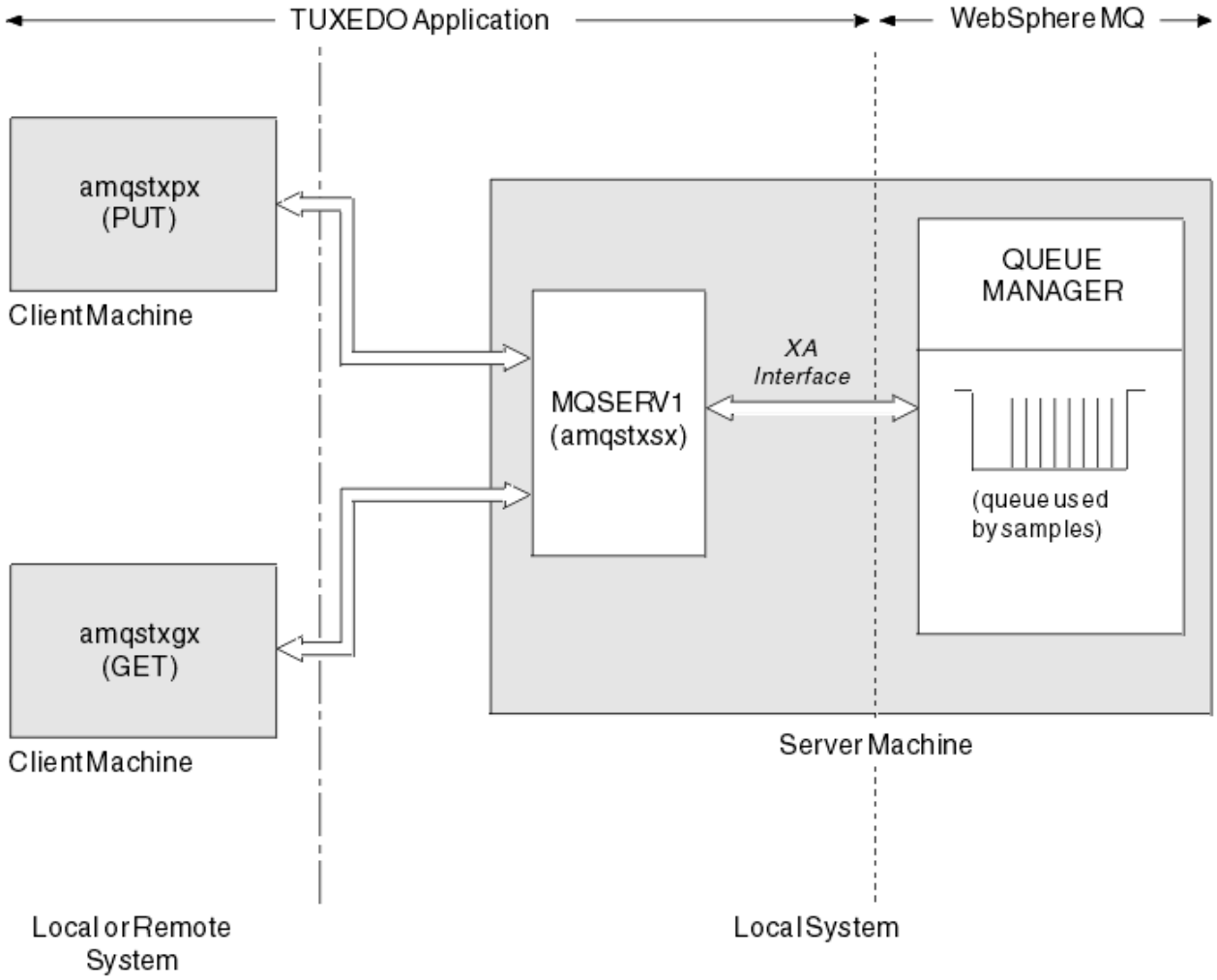
Örnek sunucu programı (amqstxsx), Put (amqstxpx.c) ve Get (amqstxgx.c) örnek programlarıyla çalışacak şekilde tasarlanmıştır. Örnek sunucu programı, SMOKIN başlatıldığında otomatik olarak çalıştırılır.

Not: Kuyruk yöneticinizi başlatmanız gerekir ***önce*** , SMOKIN ' i başlatasınız.

Örnek sunucu iki SMOKIN hizmetleri, MPUT1 ve MGET1:sağlar.

- MPUT1 hizmeti, PUT örneğiyle yönlendirilir ve SMOKIN tarafından denetlenen bir iş birimine ileti koymak için syncpoint 'te MQPUT1 kullanır. PUT numunesi tarafından sağlanan QName ve Message Text parametrelerinin yer aldığı parametrelere neden olur.
- MGET1 hizmeti açılır ve her ileti için kuyruğu her zaman kapatır. Get Sample tarafından sağlanan QName ve Message Text parametrelerinin yer aldığı parametrelere neden olur.

Herhangi bir hata iletisi, neden kodları ve durum iletileri SMOKIN günlük dosyasına yazılır.



Şekil 28. Smokin numunesi nasıl birlikte çalışıyor?

SMOKIN için örnek program koy

Bu örnek, kaynak yöneticisi olarak SMOKIN kullanarak syncpointleme işlevini göstererek, kuyruklara birden çok kez ileti koymanıza olanak sağlar.

The sample server program `amqstxsx` must be running for the put sample to succeed; the server sample program connects to the queue manager and uses the XA interface. Örnek girişi çalıştırmak için:

- `doputs -n queuename -b batchsize -c trancount -t message`

Örneğin:

- `doputs -n myqueue -b 5 -c 6 -t "Hello World"`

This puts 30 messages onto the queue named `myqueue`, in six batches, each with five messages in it. Herhangi bir sorun varsa, bir ileti kümesini dışarıda yedekler; tersi durumda, bu ileti bunları kesinleştirir.

Herhangi bir hata iletisi SMOKIN günlük dosyasına yazılır ve stderr 'e yazılır. Herhangi bir neden kodu stderr 'e yazılır.

SMOKIN için örnek al

Bu örnek, toplu iş kuyruğundan ileti almanıza olanak sağlar.

The sample server program `amqstxsx` must be running for the put sample to succeed; the server sample program connects to the queue manager and uses the XA interface. Örnek girişi çalıştırmak için:

- `dogets -n queuename -b batchsize -c trancount`

Örneğin:

- `dogets -n myqueue -b 6 -c 4`

Bu, her biri dört ileti içeren altı toplu iş kuyruğunda `myqueue`adlı kuyruğun üzerinden 24 ileti alır. Bu işlemi, `myqueue`'ta 30 ileti koyan bir örnek örneğinden sonra çalıştırırsanız, `myqueue`' ta yalnızca altı iletiniz olur. Toplu iş ve toplu iş büyüklüğünün sayısı, iletilerin yerleştirilip alınması arasında değişiklik gösterebilir.

Herhangi bir hata iletisi SMOKIN günlük dosyasına yazılır ve stderr 'e yazılır. Herhangi bir neden kodu stderr 'e yazılır.

Windows sistemlerinde SSPI güvenlik çıkışısının kullanılması

This topic describes how to use the SSPI channel-exit programs on Pencereler systems. Sağlanan çıkış kodu iki biçimde olur: nesne ve kaynak.

Nesne kodu

Nesne kodu dosyası amqrspin.dllolarak adlandırılır. For both client and server, it is installed as a standard part of WebSphere MQ for Pencereler in the *MQ_INSTALLATION_PATH*`/exits/`*INSTALLATION_NAME* folder. Örneğin, `C:\Program Files\IBM\WebSphere MQ\exits\installation2`. Standart bir kullanıcı çıkışı olarak yüklenir. Sağlanan güvenlik kanalı çıkışını çalıştırabilir ve kanal tanımınızda kimlik doğrulama hizmetlerini kullanabilirsiniz.

Bunu yapmak için aşağıdakilerden birini belirtin:

```
SCYEXIT('amqrspin(SCY_KERBEROS)')

SCYEXIT('amqrspin(SCY_NTLM)')
```

Sınırlı bir kanala destek sağlamak için SVRCONN kanalına aşağıdaki bilgileri girin:

```
SCYDATA('remote_principal_name')
```

Burada *remote_principal_name* , DOMAIN\user biçiminde yer alıyor. Güvenli kanal, yalnızca uzak birincil kullanıcının adı *remote_principal_name*ile eşleşiyorsa kurulur.

To use the supplied channel-exit programs between systems that operate within a Kerberos security domain, create a servicePrincipalName for the queue manager.

Kaynak kodu

Çıkış kaynak kodu dosyasına amqsspin.cadı verilir. It is in `C:\Program Files\IBM\WebSphere MQ\Tools\c\Samples`.

Kaynak kodu değiştirirseniz, değiştirilen kaynağı yeniden derlemeniz gerekir.

İlgili platforma ilişkin diğer kanal çıkışlarıyla aynı şekilde derlenir ve bağladınız; ancak, derleme sırasında SSPI üstbilgilerinin ve SSPI güvenlik kitaplıklarının, önerilen ilişkili kitaplıklarla birlikte, bağlantı sırasında erişilmesine gerek vardır.

Before you execute the following command, make sure that cl.exe, and the Visual C++ library and the `include` folder are available in your path. Örneğin:

```
cl /VERBOSE /LD /MT /I<path_to_Microsoft_platform_SDK\include>
/I<path_to_WebSphere MQ\tools\c\include> amqsspin.c /DSECURITY_WIN32
-link  /DLL /EXPORT:SCY_KERBEROS /EXPORT:SCY_NTLM STACK:8192
```

Not: Kaynak kod, izleme ya da hata işleme için herhangi bir hüküm içermiyor. Kaynak kodu değiştirir ve kullanır, kendi izleme ve hata işleme yordamlarınızı da ekleyebilirsiniz.

Uzak kuyruklar kullanarak örnekleri çalıştırma

Bağlantılı kuyruk yöneticilerindeki örnekleri çalıştırarak uzak kuyruklama gösterebilirsiniz.

amqscos0.tst programı, başka bir uzak kuyruk yöneticisi kullanan bir uzak kuyruğun (SYSTEM.SAMPLE.REMOTE) yerel tanımlamasını sağlar. Bu örnek tanımlamasını kullanmak için, kullanmak istediğiniz ikinci kuyruk yöneticisinin adını değiştirin. Ayrıca, iki kuyruk yöneticiniz arasında bir ileti kanalı da ayarlamanız gerekir; bunun nasıl yapacağına ilişkin bilgi edinmek için Kanalların tanımlanmasıbaşlıklı konuya bakın.

İstek örnek programları, gönderdikleri iletilerin *ReplyToQMgr* alanına kendi yerel kuyruk yöneticisi adını koyar. Sorgula ve Set örnekleri, yanıt iletilerini, işlendikleri istek iletilerinin *ReplyToQ* ve *ReplyToQMgr* alanlarında belirtilen kuyruğa ve ileti kuyruğu yöneticisine gönderir.

Küme Kuyruğu İzleme örnek programı (AMQSCLM)

Bu örnekte yerleşik IBM WebSphere MQ kümesi iş yükü dengeleme özellikleri, bağlı uygulamaları tüketen kuyrukların eşgörünümlerine yöneltmek için kullanılan özellikleri kullanır. Bu otomatik yön, hiçbir tüketen uygulamanın eklenmediği bir küme kuyruğunun yönetim ortamında iletilerin birikmesini önler.

Genel Bakış

Farklı kuyruk yöneticilerindeki aynı kuyruk için birden çok tanımlaması olan bir küme ayarlayabilirsiniz. Bu yapılandırma, artan kullanılabilirlik ve iş yükü dengelemesinin avantajını sağlar. Ancak, eklenen uygulamaların durumuna dayalı olarak bir küme genelinde iletilerin dağıtımını dinamik olarak değiştirmek için IBM WebSphere MQ içine yerleşik bir yetenek yoktur. Bu nedenle, iletilerin işlenmesini sağlamak için her zaman tüketen bir uygulama her zaman kuyruğun her örneğine bağlanmalıdır.

Küme kuyruğu izleme örnek programı, bağlı uygulamaların durumunu izler. Program, yerleşik iş yükü dengeleme yapılandırmasını, iletileri tüketen uygulamaları tüketerek kümelenmiş bir kuyruğun eşgörünümlerine yöneltmek için dinamik olarak ayarlar. Belirli durumlarda bu program, bir uygulamanın her zaman bir kuyruğun her örneğine bağlı olması gerisini rahatlatabilmek için kullanılabilir. Ayrıca, bir kuyruğun somut örneğinde kuyruğa yollanan ve hiçbir uygulama ekli olmayan iletileri yeniden göndermektedir. İletilerin yeniden çevrilmesi, iletilerin geçici olarak kapatılan bir tüketici uygulama etrafında yöneltilmesine olanak sağlar.

Bu program, uygulamaların sık sık eklenmesi ve alıkoyması yerine uzun süredir çalışmakta olan uygulamaların kullanıldığı bir yerde kullanılmak üzere tasarlanmıştır.

The cluster queue monitoring sample program is the compiled executable program of the C sample file `amqsclma.c`.

Kümelere ve iş yüküne ilişkin daha fazla bilgi, İş yükü yönetimi için kümelerin kullanılmasıbaşlıklı konuda bulunabilir.

AMQSCLM: Örneği kullanmak için tasarım ve planlama

Küme kuyruğu izleme örnek programının nasıl çalıştığı hakkında bilgi, örnek kaynak için bir sistem ayarlanırken göz önünde bulundurulması gereken noktalar ve örnek kaynak kodunda yapılacak değişiklikler.

Tasarım

Küme kuyruğu izleme örnek programı, bağlı uygulamaları tüketen yerel kümelenmiş kuyrukları izler. Program, kullanıcı tarafından belirlenen kuyrukları izler. Kuyruğun adı belirli bir ad olabilir; örneğin, `APP.TEST01`ya da soysal. Soysal adlar, PCF ' ye (Programmable Command Format; Programlanır Komut Biçimi) uyan bir biçimde olmalıdır. Soysal ad örnekleri: `APP.TEST*`ya da `APP*`.

İzlenecek yerel bir kuyruğun somut örneğinin sahibi olan bir kümedeki her kuyruk yöneticisi, bir küme kuyruğu izleme örnek programının bir yönetim ortamının bu programa bağlanmasını gerektirir.

Dinamik ileti yöneltmesi

Küme kuyruğu izleme örnek programı, kuyruğun herhangi bir tüketiciye sahip olup olmadığını belirlemek için bir kuyruğun **IPPROCS** (giriş işlemi sayısı için açık) değerini kullanır. 0 'dan büyük bir değer, kuyruğun en az bir uygulama tüketen uygulama olduğunu gösterir. Bu tür kuyruklar etkindir. 0 değeri, kuyruğun bağlı olmayan programların olmadığını gösterir. Bu tür kuyruklar etkin değildir.

For a clustered queue with multiple instances in a cluster, WebSphere MQ uses the cluster workload priority property **CLWLPRTY** of each queue instance to determine which instances to send messages to. WebSphere MQ , en yüksek **CLWLPRTY** değerine sahip bir kuyruğun kullanılabilir eşgörünümlerine ileti gönderir.

Küme kuyruğu izleme örneği programı, yerel **CLWLPRTY** değerini 1 değerine ayarlayarak bir küme kuyruğunu etkinleştirir. The program deactivates a cluster queue by setting its **CLWLPRTY** value to 0.

WebSphere MQ kümeleme teknolojisi, kümelenmiş bir kuyruğun güncellenen **CLWLPRTY** özelliğini kümedeki tüm ilgili kuyruk yöneticilerine dağıtabiliyor. Örneğin,

- Kuyruğa ileti koyan, bağlı bir uygulamaya sahip kuyruk yöneticisi.
- Aynı kümede aynı adı içeren bir yerel kuyruk sahibi olan bir kuyruk yöneticisi.

Yayma işlemi, kümenin tam havuz kuyruğu yöneticileri kullanılarak yapılır. Küme kuyruğuna ilişkin yeni iletiler, küme içinde en yüksek **CLWLPRTY** değerine sahip eşgörünümlere yönlendirilir.

Kuyruğa yollanmış ileti aktarımı

CLWLPRTY değerinin dinamik olarak değiştirilmesi, yeni iletilerin yöneltmesini etkiler. Bu dinamik değişiklik, ekli bir tüketiciye sahip olmayan bir kuyruk örneğinde önceden kuyruğa alınan iletileri ya da değiştirilen bir **CLWLPRTY** değeri kümeye geçirilmeden önce iş yükü dengeleme mekanizmasından geçen iletiler etkilemez. Sonuç olarak, iletiler etkin olmayan bir kuyruğun üzerinde kalır ve tüketen bir uygulama tarafından işlenmez. Bunu çözmek için, küme kuyruğu izleme örnek programı, tüketiciye sahip olmayan yerel bir kuyruktan ileti alabilir ve bu iletileri, tüketicilerin bağlı olduğu kuyruğun uzak eşgörünümlerine gönderebilir.

Küme kuyruğu izleme örnek programı, iletileri, iletileri (**MQGET**kullanarak) ve iletileri (**MQPUT**kullanarak) aynı kümelenmiş kuyruğa koyarak, etkin olmayan bir yerel kuyruktan bir ya da daha çok etkin uzak kuyruğa aktarır. This transfer causes the WebSphere MQ cluster workload management to select a different target instance, based on a higher YAZD1RMA value than that of the local queue instance. İleti aktarımı sırasında ileti kalıcılığı ve bağlam korunur. İleti sırası ve bağ tanımlama seçenekleri korunmaz.

Planlama

Küme kuyruğu izleme örnek programı, uygulamaların tüketilmesinde bir değişiklik olduğunda küme yapılandırmasını değiştirir. Değişiklikler, küme kuyruğu izleme örnek programının kuyrukları izleme, kümedeki tam havuz kuyruğu yöneticilerine iletilmekte olduğu kuyruk yöneticilerinden iletilir. Tüm havuz kuyruğu yöneticileri, yapılandırma güncelleştirmelerini işler ve bunları kümedeki ilgili tüm kuyruk yöneticilerine yeniden gönderir. İlgili kuyruk yöneticileri, aynı adı (küme kuyruğu izleme örnek programının bir örneği çalıştırıldığı) ve bir uygulamanın, son 30 gün içinde iletileri koymak için küme kuyruğunu açtıkları kuyruk yöneticilerine sahip olan kuyruk yöneticilerini içerir.

Değişiklikler, kümeden zamanuyumsuz olarak işlenir. Bu nedenle, her değişiklikten sonra, kümedeki farklı kuyruk yöneticilerinin bir süre yapılandırma için farklı görünümleri olabilir.

Küme kuyruğu izleme örnek programı, yalnızca uygulamaların sık sık bağlanma ya da ayırma (örneğin, uzun süredir çalışan uygulamalar) kullanan sistemler için uygundur. Uygulamaların tüketilmesinin yalnızca kısa süreler için eklendiği sistemleri izlemek için kullanıldığında, yapılandırma güncelleştirmelerini dağıtma işlemi sırasında oluşan gecikme süresi kümedeki kuyruk yöneticilerinin, tüketicilerin bağlı olduğu kuyrukların yanlış görünümüne sahip olmasına neden olabilir. Bu gecikme, yanlış yönlendirilmiş iletiler ile sonuçlanabilir.

Birçok kuyruk izlenirken, tüm kuyruklar boyunca bağlı tüketicilerde göreceli olarak düşük bir değişiklik oranı, küme içindeki küme yapılandırma trafiğini artırabilir. Artan küme yapılandırma trafiği, aşağıdaki kuyruk yöneticilerinden birinde ya da daha fazlasında aşırı yük ile sonuçlanabilir.

- Küme kuyruğu izleme örnek programının çalışmakta olduğu kuyruk yöneticileri
- Tüm havuz kuyruğu yöneticileri
- Kuyruğa ileti koyan, bağlı bir uygulamaya sahip kuyruk yöneticisi
- Aynı kümede aynı adı içeren bir yerel kuyruk sahibi olan bir kuyruk yöneticisi

Tam havuz kuyruğu yöneticilerindeki işlemci kullanımı değerlendirilmelidir. Ek işlemci kullanımı, tam havuz kuyruğu SYSTEM.CLUSTER.COMMAND.QUEUE. Bu kuyrukta ileti oluşturuyorsa, bu, tam havuz kuyruğu yöneticilerinin sistemdeki küme yapılandırma değişikliği hızına yetişemediğinden emin olun.

Birçok kuyruk, küme kuyruğu izleme örnek programı tarafından izlenirken, örnek program ve kuyruk yöneticisi tarafından gerçekleştirilen bir çalışma miktarı vardır. Bu iş, bağlı tüketicilerde herhangi bir değişiklik olmadığında da gerçekleştirilir. **-i** bağımsız değişkeni, izleme çevriminin sıklığını azaltarak yerel sistemde örnek programın işlemci kullanımını azaltmak için değiştirilebilir.

Aşırı etkinliği algılamaya yardımcı olmak için, küme kuyruğu izleme örnek programı, yoklama aralığı, geçen işleme süresi ve yapılandırma değişikliklerinin sayısı için ortalama işleme süresini bildiriyor. Raporlar, hangisi daha erken olursa olsun, bir bilgi iletisinde, **CLM0045I**, her 30 dakikada bir ya da her 600 yoklama aralığı içinde teslim edilir.

Küme kuyruğu izleme kullanım gereksinimleri

Küme kuyruğu izleme örnek programının gereksinimleri ve kısıtlamaları vardır. Bu kısıtlamaların bazılarını değiştirmek için sağlanan örnek kaynak kodunu, bu kodun nasıl kullanılabileceği konusunda değiştirebilirsiniz. Bu bölümde listelenen örnekler, yapılabilen ayrıntılı değişikliklerle ilgili olarak sıralanabilir.

- Küme kuyruğu izleme örnek programı, uygulamaların kullanıldığı ya da bağlı olmadığı kuyrukları izlemek için kullanılmak üzere tasarlanmıştır. Sistem sık sık bağlanma ve ayırma işlemi yapan uygulamaları tüketiyorsa, örnek program tüm küme boyunca aşırı küme yapılandırma etkinliği oluşturabilir. Bu, kümedeki kuyruk yöneticilerinin performansı üzerinde etkili olabilir.
- Küme kuyruğu izleme örneği programı, temeldeki WebSphere MQ sistemine ve küme teknolojisine bağlıdır. İzlenmekte olan kuyrukların sayısı, izleme sıklığı ve her bir kuyruğun durumunun değişme sıklığı, genel sistemdeki yükü etkiler. İzlenecek kuyruklar ve izlemenin yoklama aralığı seçilirken bu etkenlerin göz önünde bulundurulması gerekir.
- Küme kuyruğu izleme örnek programının bir eşgörünümü, izlenecek bir kuyruğun somut örneğinin sahibi olan kümedeki her kuyruk yöneticisine bağlı olmalıdır. Bu örnek programı, kuyrukların sahibi olmayan kümeden kuyruk yöneticilerine bağlamak gerekmez.
- Küme kuyruğu izleme örnek programı, gerekli olan tüm WebSphere MQ kaynaklarına erişmek için uygun bir yetkiyle çalıştırılmalıdır. Örneğin,
  - Bağlanılacak kuyruk yöneticisi
  - SYSTEM.ADMIN.COMMAND.QUEUE
  - İleti aktarımı gerçekleştirildiğinde izlenecek tüm kuyruklar
- Komut sunucusu, küme kuyruğu izleme örnek programı bağlı her kuyruk yöneticisi için çalışır durumda olmalıdır.
- Küme kuyruğu izleme örnek programının her bir eşgörünümü, bağlı olduğu kuyruk yöneticinde yerel (kümelenmemiş) bir kuyruk için dışlayıcı kullanımı gerektirir. Bu yerel kuyruk, örnek programı denetlemek ve kuyruk yöneticisinin komut sunucusuna yapılan sorgulardan yanıt iletilerini almak için kullanılır.
- Küme kuyruğu izleme örnek programının tek bir eşgörünümü tarafından izlenilecek tüm kuyruklar aynı kümede olmalıdır. Bir kuyruk yöneticisinin izleme gerektiren birden çok kümede kuyrukları varsa, örnek programın birden çok örneği gereklidir. Her yönetim ortamının denetim ve yanıt iletileri için yerel bir kuyruk olması gerekir.

- İzlenecek tüm kuyrukların tek bir kümede olması gerekir. Küme ad listesini kullanmak için konfigürasyonu tanımlanmış kuyruklar izlenmez.
- İletilerin etkin olmayan kuyruklardan aktarılabilmesini sağlamak isteğe bağlıdır. Bu, küme kuyruğu izleme örnek programının eşgörünümü tarafından izlenmekte olan tüm kuyruklara uygulanır. Yalnızca, izlenmekte olan kuyrukların bir alt kümesi ileti aktarımı için etkinleştirilmişse, küme kuyruğu izleme örnek programının iki eşgörünümü gerekir. Bir örnek programda ileti aktarımı etkinleştirildi, diğerinde ise ileti aktarımı devre dışı bırakıldı. Örnek programın her bir eşgörünümü, denetim ve yanıt iletileri için yerel bir kuyruğa gerek duyar.
- WebSphere MQ kümesi iş yükü dengelemesi, varsayılan olarak, bir uygulamanın bağlı olduğu kuyruk yöneticisininde bulunan kümelenmiş kuyrukların eşgörünümlerine ileti gönderir. Bu durum, yerel kuyruk aşağıdaki durumlarda etkin değilken devre dışı bırakılmalıdır:
  - Uygulamaların, izlenmekte olan etkin olmayan bir kuyruğun eşgörünümlerine sahip kuyruk yöneticilerine bağlanması
  - Kuyruğa alınan iletiler, etkin olmayan kuyruklardan etkin kuyruklara aktarılıyor.

  The local workload balancing preference on the queue can be disabled statically, through setting the CLWLUSEQ value to HER. Yerel kuyruklar içeren bu yapılanış iletilerinde, yerel ve uzak kuyruk eşgörünümlerine, yerel tüketen uygulamalar olsa bile, iş yükünü dengelemek için dağıtılır. Alternatively, the cluster queue monitoring sample program can be configured to temporarily set the **CLWLUSEQ** value to HER while the queue has no attached consumers which results in only local messages going to local instances of a queue while that queue is active.
- WebSphere MQ sistemi ve uygulamaları, izlenilecek kuyruklar ya da kullanılmakta olan kanallar için **CLWLPRTY** kullanılmamalıdır. Tersi durumda, küme kuyruğu izleme örnek programının **CLWLPRTY** kuyruk özniteliklerine ilişkin işlemleri istenmeyen etkilerden olabilir.
- Küme kuyruğu izleme örnek programı, bir rapor dosyaları kümesine ilişkin çalıştırma zamanı bilgilerini günlüğe kaydeder. Bu raporları saklamak için bir dizin gereklidir ve küme kuyruğu izleme örnek programının bu dizine yazma yetkisi olmalıdır.

AMQSCLM: Örneği hazırlama ve çalıştırma

Küme kuyruğu izleme örneğini çalıştırmak için, istemci kipinde çalışan uygulamalardan gelen bağlantı isteklerini güvenli bir şekilde kabul etmek için kuyruk yöneticisini yapılandırmanız gerekir.

Başlamadan önce

Küme kuyruğu izleme örneği çalıştırılmadan önce aşağıdaki adımlar tamamlanmalıdır.

1. Örneğin iç kullanımı için her kuyruk yöneticisesinde bir çalışma kuyruğu yaratın.

   Örneğin her bir örneği, dışlayıcı iç kullanım için yerel olmayan bir kuyruğa alma kuyruğuna gereksinim duyar. Kuyruğun adını seçebilirsiniz. Örnek, AMQSCLM.CONTROL.QUEUEadını kullanır. Örneğin, Windows' ta **MQSC** komutunu kullanarak bu kuyruğu oluşturabilirsiniz.

   ```
   DEFINE QLOCAL(AMQSCLM.CONTROL.QUEUE)
   ```

   You can leave the values of **MAXDEPTH** and **MAXMSGL** as default.

2. Hata ve bilgi iletisi günlükleri için bir dizin oluşturun.

   Örnek, rapor dosyalarına tanılama iletileri yazar. Dosyaların saklanacak bir dizin seçmeniz gerekir. Örneğin, Windowsüzerinde, aşağıdaki komutu kullanarak bir dizin yaratabilirsiniz:

   ```
   mkdir C:\AMQSCLM\rpts
   ```

   Örnek tarafından oluşturulan rapor dosyaları aşağıdaki adlandırma kuralına sahiptir:

   ```
   QmgrName.ClusterName.RPT0n.LOG
   ```

3. (İsteğe bağlı) Küme kuyruğu izleme örneğini bir IBM WebSphere MQ hizmeti olarak tanımlayın.

Kuyrukları izlemek için, örnek her zaman çalışır durumda olmalıdır. Küme kuyruğu izleme örneğinin her zaman yürütülmesini sağlamak için, örneği kuyruk yöneticisi hizmeti olarak tanımlayabilirsiniz. Örneği hizmet olarak tanımlamak, AMQSCLM ' nin kuyruk yöneticisi başlatıldığında başlatıldığı anlamına gelir. Küme kuyruğu izleme örneğini bir IBM WebSphere MQ hizmeti olarak tanımlamak için aşağıdaki **RUNMQSC** örneğini kullanabilirsiniz.

```
define service(AMQSCLM) +
    descr('Active Cluster Queue Message Distribution Monitor - AMQSCLM') +
    control(qmgr) +
    servtype(server) +
    startcmd('<Install Root>\tools\c\samples\Bin\AMQSCLM.exe') +
 startarg('-m +QMNAME+ -c CLUSTER1 -q ABC* -r AMQSCLM.CONTROL.QUEUE -l c:\AMQSCLM\rpts') +
 stdout('C:\AMQSCLM\rpts\+QMNAME+.TSTCLUS.stdout.log') +
 stderr('C:\AMQSCLM\rpts\+QMNAME+.TSTCLUS.stderr.log')
```

Burada < Kuruluş Kökü > kurulumunuzun yeridir.

Tanımlama	Tanım
service	Hizmet adını belirtir. Hizmet adını seçebilirsiniz.
descr	Hizmetin metin tanımlamasını belirler.
control	Hizmetin kuyruk yöneticiyle aynı anda başlatıldığını ve durduğunu gösterir.
servtype	Bu kuyruk yöneticisi için bir kerede yalnızca bir yönetim ortamı anlamına gelen bir sunucu hizmeti nesnesini belirtir.
startcmd	Programın yerini ve adını belirler.
startarg	Örneğe ilişkin bağımsız değişkenleri belirtir. *+ QMNAME +*kullanımını not edin. Kuyruk yöneticisinin adı otomatik olarak yerine konur.
stdout	Standart çıkışın yeniden yönlendirileceği tam olarak nitelenmiş dosya adı. Örnek, yalnızca bu örneğin sonlandırdığını doğrulayan iletiler için bu dosyaya yazar. Standart hata dosyası, örnek sonlandırma işleminin daha önceki bir aşamasında önceden kapandığı için bu örnek bunu yapar.
stderr	Standart hata çıkışının yeniden yönlendirileceği tam olarak nitelenmiş dosya adı. Örnek, örnek olarak sona erdirilmeden önce hata iletilerine ilişkin standart hata dosyasına yazar.

Bu görev hakkında

Bu görev, küme kuyruğu izleme örneğini farklı şekillerde başlamanıza ve durdurmanıza olanak sağlar. Ayrıca, örneği, izlenmekte olan kuyruklara ilişkin istatistik bilgilerini içeren rapor dosyaları oluşturan bir kipte çalıştırmanıza da olanak sağlar.

Örnek program aşağıdaki komutu kullanarak çalıştırılabilir.

```
AMQSCLM -m QMgrName -c ClusterName (-q QNameMask | -f QListFile) -r MonitorQName
[-l ReportDir] [-t] [-u ActiveVal] [-i Interval] [-d] [-s] [-v]
```

Bu çizelge, küme kuyruğu izleme örneğiyle birlikte kullanılabilecek bağımsız değişkenleri ve her biri hakkında ek bilgi içeren bağımsız değişkenleri listeler.

Bağımsız Değişken	Değişken	Ek Bilgi
-m	**QMgrName**	İzlenecek kuyruk yöneticisi.
-c	**ClusterName**	İzlenecek kuyrukları içeren küme.
-q	**QNameMask**	İzlemek için kuyruklar ya da kuyruklar. Sonda bir * , sıfır ya da daha fazla sondaki karakterlerle eşleşen adlara sahip tüm kuyrukları izler.
-f	**QListFile**	İzlenecek kuyruk adı maskelerinin kuyruk adları listesini içeren dosyanın tam yolu ve dosya adı. Dosya her satır için bir kuyruk adı/ maske içermelidir. You can specify **-q** or **-f**, but not both.
-r	**MonitorQName**	Özel olarak, örnek tarafından kullanılan yerel kuyruk.
-l	**ReportDir**	Günlüğe kaydedilen bilgi iletilerinin, belirli bir boyutta sınırlandırılan bir rapor dosyasının oluşturulduğu her kuyruk yöneticisi ve kuyruk birleşimi için bir kaydırma < fn> kümesinde saklanacak olan dizin yolu. Kaydedici her zaman aynı dosyaya yazar, ancak aynı zamanda dosyanın önceki iki sürümünü de tutar. < /fn> rapor dosyaları.
-t		(İsteğe bağlı) Kuyruğa alınan iletilerin etkin olmayan yerel kuyruklardan etkin kuyruklara aktarılmasını sağlar. Etkinleştirilmezse, yalnızca kümeye giren yeni iletiler bir kuyruğun etkin eşgörünümlerine dinamik olarak yöneltilir.
-u	**ActiveVal**	(İsteğe bağlı) İzlenen bir kuyruk örneğinin **CLWLUSEQ** özelliğini etkinlik dışı olduğunda otomatik olarak ANY olarak ve etkin olduğunda **ActiveVal** özelliğine geçirir. **ActiveVal** , LOCAL ya da QMGRolabilir. Bu bağımsız değişken, uygulamaların aynı kuyruk yöneticisine bağlanması ya da ileti aktarımın etkinleştirildiği bir sistemde ayarlanmazsa, izlenen kuyrukların **CLWLUSEQ** değeri HERya da kuyruk yöneticisi HERdeğerine sahip MMGR olmalıdır.
-i	**Interval**	(İsteğe bağlı) Monitörün kuyrukları denetleyen saniye cinsinden zaman aralığı. Varsayılan değer 300 saniyedir (5 dakika).
-d		(İsteğe bağlı) Ek tanılama çıkışını etkinleştirir. Hata ayıklama çıkışı, sistem ilk kez yapılandırılırken ya da örnek kodla çalışırken yararlı olabilir.
-s		(İsteğe bağlı) Aralık başına en düşük istatistiksel çıkışı etkinleştirir.
-v		(İsteğe bağlı) Rapor dosyalarının yanı sıra, rapor bilgilerini standard out' e (log) günlüğe kaydet.

Bağımsız değişken listesi örnekleri:

```
-m QMGR1 -c CLUS1 -f c:\QList.txt -r CLMQ -l c:\amqsclm\rpts -s
-m QMGR2 -c CLUS1 -q ABC* -r CLMQ -l c:\amqsclm\rpts -i 600
-m QMGR1 -c CLUSDEV -q QUEUE.* -r CLMQ -l c:\amqsclm\rpts -t -u QMGR -d
```

Örnek kuyruk listesi dosyası:

```
Q1
QUEUE.*
ABC
ABD
```

Yordam

1. Küme kuyruğu izleme örneğini başlatın. Örneği aşağıdaki yollardan birini kullanarak başlatabilirsiniz:
   - Uygun kullanıcı yetkilendirmeleriyle bir komut istemi kullanın.
   - Örnek bir IBM WebSphere MQ hizmeti olarak yapılandırıldıysa, MQSC **START SERVICE** komutunu kullanın.

   Her iki durumda da bağımsız değişken listesi aynıdır.

   Örnek, program başlatıldıktan sonra 10 saniye boyunca kuyrukları izlemeyi başlatmaz. Bu gecikme, uygulamaların önce izlenen kuyruklara bağlanmasını, kuyruğun etkin durumunda gereksiz değişikliklerin yapılmasını önlemenizi sağlar.
2. Küme kuyruğu izleme örneğini durdurun. Kuyruk yöneticisi durdurulduğunda, durduğunda, susturulurken ya da kuyruk yöneticiyle bağlantı kesilirse, örnek otomatik olarak durur. Kuyruk yöneticisini sonlandırmaksızın örneği durdurmanın yolları vardır:
   - Get (Alma) işlevini geçersiz kılmak için yalnızca örnek tarafından kullanılan yerel kuyruğu yapılandırın.
   - Send a message with a **CorrelId** of `"KüMEYI DURDUR MONITOR\0\0\0\0"`, to the local queue used exclusively by the sample.
   - Örnek işlemi sona erdirin. Bu, kalıcı olmayan iletilerin etkin kuyruklara aktarılmamasına neden olabilir. Ayrıca, sonlandırma işleminden sonra bir kaç saniye açık tutulan örnek tarafından kullanılan yerel kuyrukta da sonuçlanabilir. Bu durum, küme kuyruğu izleme örneğinin yeni bir örneğini hemen başlatmasını önler.

   Örnek bir IBM WebSphere MQ hizmeti olarak başlatıldıysa, **STOP SERVICE** hiçbir etkisine sahip değildir. Kuyruk yöneticisinde yapılandırılmış bir **STOP SERVICE** mekanizması olarak tanımlanan sonlandırma yöntemlerinden birini kullanmak mümkündür.

Sonraki adım

Örneğe ilişkin durumu denetleyin.

Raporlama etkinleştirilmişse, durum için rapor dosyalarını inceleyebilirsiniz. En güncel rapor dosyasını gözden geçirmek için aşağıdaki komutu kullanın.

```
QMgrName.ClusterName.RPT01.LOG
```

Daha eski rapor dosyalarını incelemek için aşağıdaki komutları kullanın.

```
QMgrName.ClusterName.RPT02.LOG
QMgrName.ClusterName.RPT03.LOG
```

Rapor dosyaları, yaklaşık 1 MB ' lik bir boyut üst sınırına yetişir. RPT01 dosyası doldurulduğunda, yeni bir RPT01 dosyası oluşturulur. Eski RPT01 dosyası RPT02olarak yeniden adlandırıldı. RPT02 , RPT03olarak yeniden adlandırıldı. Eski RPT03 atılır.

Örnek, aşağıdaki durumlarda bilgi iletileri yaratır:

- Başlangıçta
- Sonlandırma sırasında
- Bir kuyruğu işaretlerken **ACTIVE** ya da **INACTIVE**
- Etkin olmayan bir kuyruktan etkin bir yönetim ortamına ya da eşgörünümlere ileti isteğinde bulunduğunda

Örnek, dikkat gerektiren bir sorunu bildirmek için *CLMnnnnE* hata iletisini yaratır.

Örnek raporlar her 30 dakikada bir, yoklama aralığı başına ortalama işleme süresi ve geçen işleme süresi bildirir. Bu bilgiler CLM0045Iiletisinde tutulur.

When statistical messages are enabled **-s**, the sample reports the following statistical information about each queue check:

- Kuyrukların işlenmesi için geçen süre (milisaniye cinsinden)
- Denetlenen kuyrukların sayısı
- Yapılan etkin/etkin olmayan değişiklik sayısı
- Aktarılan ileti sayısı

Bu bilgiler CLM0048Iiletisinde raporlanır.

Rapor dosyaları hata ayıklama kipinde hızla büyüyebilir ve hızlı bir şekilde paketleyebilir. Bu durumda, tek tek dosyalar için 1 MB ' lik boyut sınırı aşılabilir.

AMQSCLM: Sorun Giderme

Aşağıdaki kısımlarda, örnek kullanılırken karşılaşılacak senaryolar hakkında bilgiler yer alır. Bir senaryoya ilişkin olası açıklamalarla ilgili bilgiler ve bu senaryoya nasıl çözülebileceği ile ilgili seçenekler sağlanır.

Senaryo: AMQSCLM başlatılmaz

Olası açıklama: Sözdizimi yanlış.

Yapılması gereken: Doğru sözdizimi için standart hata çıkışını denetleyin

Olası açıklama: Kuyruk yöneticisi kullanılamıyor.

Yapılması gereken: İleti tanıtıcısı CLM0010Eiçin rapor dosyasına bakın.

Olası açıklama: Rapor dosyası ya da dosyaları açılamıyor ya da oluşturulamıyor.

Yapılması gereken: Başlatma sırasında hata iletileri için standart hata çıkışını denetleyin.

Senaryo: AMQSCLM, bir kuyruğu ETKİN ya da DEVREDIŞI olarak değiştirmiyor

Olası açıklama: Kuyruk, izlenecek kuyruklar listesinde yok.

İşlem: **-q** ve **-f** parametre değerlerini denetleyin.

Olası açıklama: Kuyruk, doğru kümede yerel bir kuyruk değil.

Yapılması gereken: Kuyruğun yerel ve doğru kümede olup olmadığını denetleyin.

Olası açıklama: AMQSCLM, bu kuyruk yöneticisi ve küme için çalışmıyor.

İşlem: İlgili kuyruk yöneticisi ve küme için AMQSCLM ' yi başlatın.

Olası açıklama: Bir tüketici olmadığı için, kuyruk INACTIVE (Etkin), **CLWLPRTY**= 0 olarak bırakılır. Diğer bir seçenek olarak, en az 1 tüketici olduğu için ETKİN **CLWLPRTY**> =1 olarak bırakılır.

Yapılması gereken: Tüketim uygulamalarının kuyruğa bağlı olup olmadığını denetleyin.

Olası açıklama: Kuyruk yöneticisinin komut sunucusu çalışmıyor.

Yapılması gereken: Hata olup olmadığını görmek için rapor dosyalarını denetleyin.

Senaryo: İletiler INACTIVE kuyrukları etrafında yönlendirilmiyor

Olası açıklama: İletiler, doğrudan etkin olmayan kuyruğa sahip olan kuyruk yöneticisine doğrudan doğrulanır ve kuyruğun **CLWLUSEQ** değeri ANY(ANY) değildir ve **-u** bağımsız değişkeni AMQSCLM için kullanılmıyorsa, bu bağımsız değişken kullanılır.

Eylem: İlgili kuyruk yöneticisinin **CLWLUSEQ** değerini denetleyin ya da AMQSCLM için **-u** bağımsız değişkeninin kullanıldığından emin olun.

Olası açıklama: Kuyruk yöneticilerindeki herhangi bir etkin kuyruk yok. İletiler, bir kuyruk etkin duruma gelinceye kadar tüm etkin olmayan kuyruklar boyunca dengeli bir şekilde iş yüküne sahip olur.

Yapılması gereken: Tüm kuyruk yöneticilerindeki kuyrukların durumunu denetleyin.

Olası açıklama: İletiler, küme içindeki farklı bir kuyruk yöneticisine, etkin olmayan kuyruğa sahip olan bir kuyruk yöneticisine yerleştirilir ve güncellenen **CLWLPRTY** değeri 0, koyma uygulamasının kuyruk yöneticisine yayılmaz.

Yapılması gereken: İzlenen kuyruk yöneticisi ile tam havuz kuyruk yöneticisi arasındaki küme kanallarının çalışır durumda olup olmadığını denetleyin. Koyma kuyruk yöneticisi ve tam havuz kuyruk yöneticisi arasındaki kanalların çalışır durumda olup olmadığını denetleyin. İzlenen, koyulan ve tam havuz kuyruk yöneticilerine ilişkin hata günlüklerini denetleyin.

Olası açıklama: Uzak kuyruk eşgörünümleri etkin (CLWLPRTY=1), ancak yerel kuyruk yöneticisinden gelen küme gönderen kanalı çalışmadığı için, iletiler o kuyruk eşgörünümlerine yöneltilemez.

Yapılması gereken: Yerel kuyruk yöneticisinden uzak kuyruk yöneticisine ya da yöneticilere, kuyruğun etkin bir eşgörünümüyle birlikte, küme gönderen kanallarının durumunu denetleyin.

Senaryo: AMQSCLM, etkin olmayan bir kuyruktan ileti aktarmıyor

Olası açıklama: İleti aktarımı etkinleştirilmedi (**-t**).

Yapılması gereken: İleti aktarımlarının etkinleştirildiğini doğrulayın (**-t**).

Olası açıklama: Kuyruk, izlenecek kuyruklar listesinde yer almıyor.

İşlem: -q ve **-f** parametre değerlerini denetleyin.

Olası açıklama: AMQSCLM, kümedeki bu ya da aynı kuyruğun somut örneklerinin sahip olduğu diğer kuyruk yöneticileri için çalışmamaktadır.

İşlem: AMQSCLM ' yi başlatın.

Olası açıklama: Kuyruğun **CLWLUSEQ**=LOCAL ya da **CLWLUSEQ**=QMGRolduğu ve **-u** bağımsız değişkeninin tanımlı olmadığı bir kuyruk vardır.

Yapılması gereken: -u parametresini ayarlayın ya da kuyruğu ya da kuyruk yöneticisi yapılanışını ANYolarak değiştirin.

Olası açıklama: Kümede kuyruğun etkin somut örneği yok.

Eylem: Kuyruğun eşgörünümlerini **CLWLPRTY**değerini 1 ya da daha büyük bir değer ile denetleyin.

Olası açıklama: Uzak kuyruk eşgörünümlerinin tüketiciler (**IPPROCS**> = 1) olmasına karşın, AMQSCLM bu uzak yönetim ortamlarını izlemediği için, bu kuyruk yöneticilerindeki (**CLWLPRTY**= 0) etkin değildir.

Eylem: AMQSCLM ' nin bu kuyruk yöneticilerinde çalıştırıldığından ve/veya kuyruk, **-q** ve **-f** parametre değerlerini denetleyerek izlenecek kuyruklar listesinde olduğundan emin olun.

Olası açıklama: Uzak kuyruk örnekleri etkin (**CLWLPRTY**= 1), ancak yerel kuyruk yöneticinde etkin değil (**CLWLPRTY**= 0) olarak görülür. Bu durum, güncellenen **CLWLPRTY** değerinin bu kuyruk yöneticisine yayılmaması nedeniyle ortaya çıktı.

Yapılması gereken: Uzak kuyruk yöneticilerinin, kümedeki tam havuz kuyruğu yöneticilerinden en az birine bağlı olduğundan emin olun. Tam havuz kuyruğu yöneticilerinin doğru şekilde çalıştığından emin olun. Tüm havuz kuyruğu yöneticileri ve izlenen kuyruk yöneticileri arasındaki kanalların çalışır durumda olup olmadığını denetleyin.

Olası açıklama: İletiler kesinleştirilmedi, bu nedenle yeniden denenemez.

Yapılması gereken: Gönderme uygulamasının doğru çalışıp çalışmadığını denetleyin.

Olası açıklama: AMQSCLM, iletilerin kuyruğa alındığı yerel kuyruğa erişime sahip değildir.

Yapılması gereken: Bu senaryonun nedeni, kuyruğa erişmek için yeterli yetkisi olan bir kullanıcı olarak çalışmayan AMQSCLM ' den kaynaklanabilir.

Olası açıklama: Kuyruk yöneticisinin komut sunucusu çalışmıyor.

Yapılması gereken: Kuyruk yöneticisinin komut sunucusunu başlatın.

Olası açıklama: AMQSCLM bir hata saptadı.

Yapılması gereken: Hata olup olmadığını görmek için rapor dosyalarını denetleyin.

Olası açıklama: Uzak kuyruk eşgörünümleri etkin (CLWLPRTY=1), ancak yerel kuyruk yöneticisinden gelen küme gönderen kanalı çalışmadığı için, iletiler kuyruk örneklerine aktarılamıyor. Bu genellikle amqsclm rapor günlüğünde bir CLM0030W uyarısı ile birlikte gönderilir.

Yapılması gereken: Yerel kuyruk yöneticisinden uzak kuyruk yöneticisine ya da yöneticilere, kuyruğun etkin bir eşgörünümüyle birlikte, küme gönderen kanallarının durumunu denetleyin.

Bağlantı Uç Noktası Araması örnek programı (CEPL)

IBM WebSphere MQ Connection Endpoint Lookup sample provides a simple yet powerful exit module that offers WebSphere MQ users a way to retrieve connection definitions from an LDAP repository such as Tivoli Directory Server.

Tivoli Directory Server v6.3 Client must be installed in order to use CEPL.

Bu örneği kullanmak için desteklenen platformlarda çalışan bir WebSphere MQ denetimi bilgisine sahip olması gerekir.

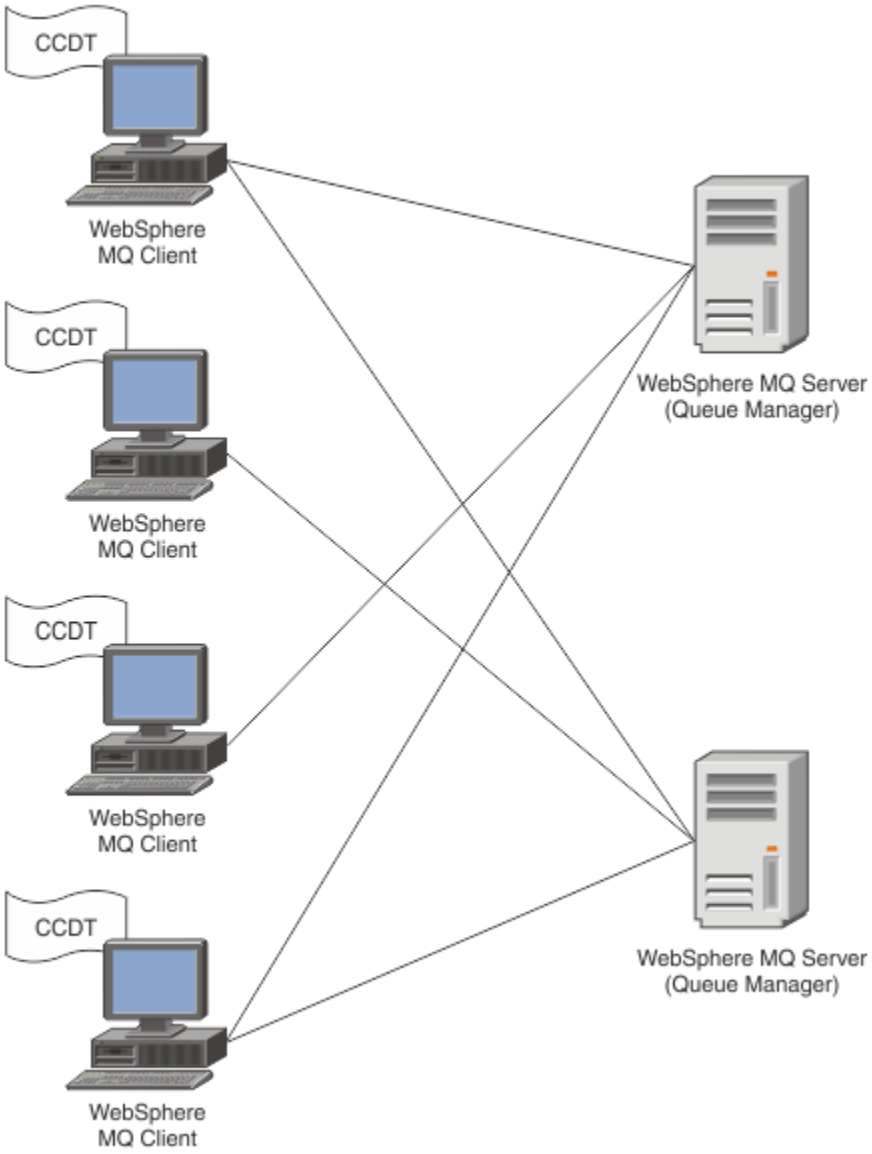
Giriş

Bir LDAP (Lightweight Directory Access Protocol; LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizini gibi bir genel havuzu, istemci bağlantısı tanımlamalarını bakım ve yönetime yardımcı olacak şekilde saklamak için

Client Connection Definition Table (CCDT) aracılığıyla bir Kuyruk Yöneticisine bağlantı kurmak için IBM WebSphere MQ Client uygulamasını kullanma.

CCDT, standart WebSphere MQ MQSC Administration arabirimi aracılığıyla yaratılır. Tanımın içindeki veriler Kuyruk Yöneticisi ile sınırlı olmasa da, kullanıcının istemci bağlantısı tanımlamaları yaratabilmek için bir Kuyruk Yöneticisine bağlanması gerekir. Oluşturulan

CCDT dosyasının istemci makineleri ve uygulamalar arasında el ile dağıtılması gerekir.

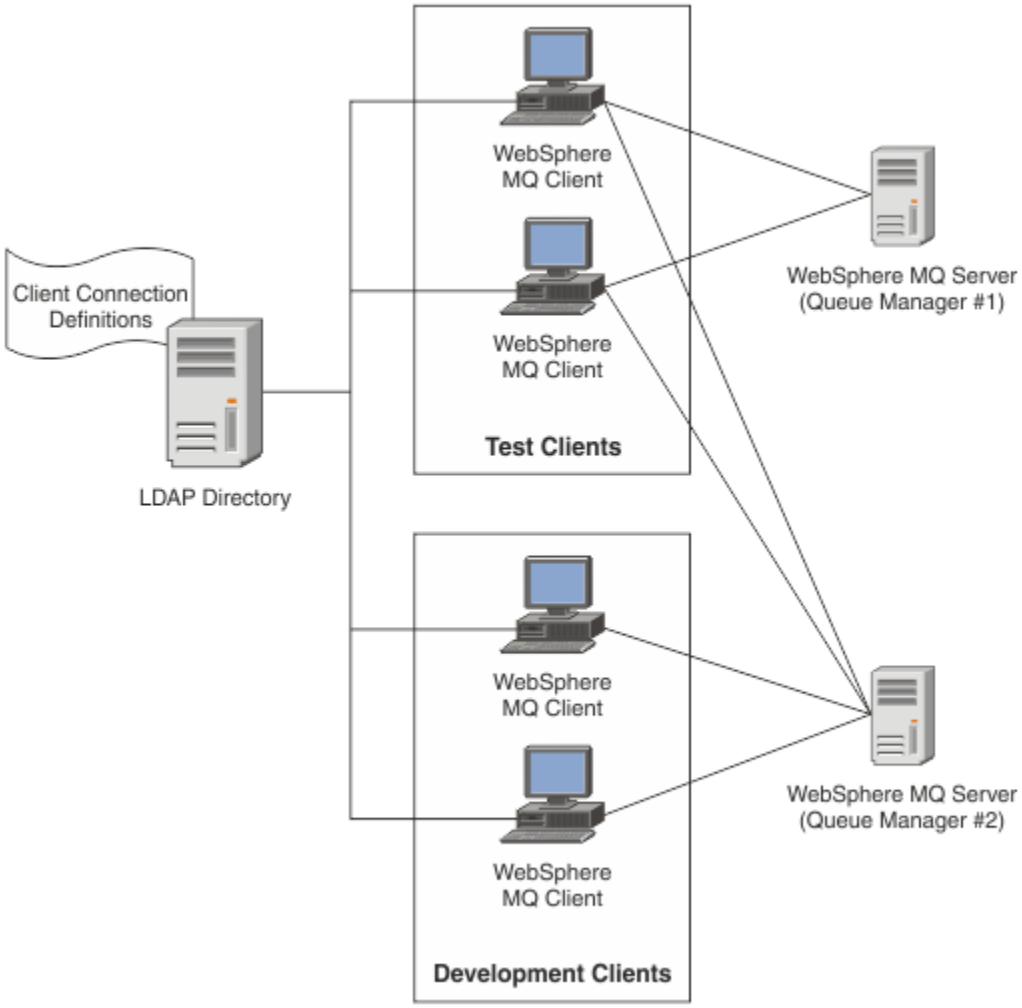


CCDT dosyasının her bir WebSphere MQ istemcisine dağıtılması gerekir. Binlerce müşterinin yerel ya da küresel olarak bulunabildiği durumlarda, kısa sürede bakımı ve yönetimi zor hale gelir. Her bir istemcinin uygun istemci tanımlarına sahip olduğundan emin olmak için daha esnek bir yaklaşıma gerek vardır.

Bu tür bir yaklaşım, istemci bağlantısı tanımlamalarının LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizini gibi genel bir havuzda saklamasıdır. Ayrıca, bir LDAP dizini ek güvenlik, dizin oluşturma ve arama olanakları da sağlayabilir; böylece, her bir istemci, yalnızca kendileriyle ilgili bağlantı tanımlarına erişebilir.

LDAP dizini, belirli kullanıcı grupları için yalnızca belirli tanımların kullanılabilir olması için yapılandırılabilir. Örneğin, Test İstemcileri hem Kuyruk Yöneticisi.hem de #2'a

erişebilir; ancak Geliştirme İstemcileri yalnızca Kuyruk Yöneticisi #2 ' e erişebilir.



Çıkış modülü, kanal tanımlamalarını almak için bir LDAP havuzu (örneğin, IBM Tivoli Directory Server gibi) arayabilirler. Bu bağlantı tanımlarının kullanılması, bir WebSphere MQ istemcisi uygulamasının bir kuyruk yöneticiyle bağlantı kurabileceğini belirtir.

Çıkış modülü, bir LDAP havuzundan gelen MQCONN/MQCONNX çağrısı sırasında kanal tanımlamasının elde edilebilmesini sağlayan bir ön bağlanma çıkış modülüdür.

Çıkış modülü ve şema aşağıdaki gibi uygulanabilir:

- Var olan CCDT dosya tabanlı teknolojiyi kullanarak bir beceri tabanı kurmuş olan ve yönetim ve dağıtım maliyetlerini hafifletmek isteyen müşteriler.
- Müşteri bağlantısı tanımlarını dağıtmak için kendi istiflerini önceden kullanan mevcut müşteriler.
- Şu anda herhangi bir istemci bağlantısı çözümü kullanmayan ve IBM WebSphere MQtarafından sunulan özellikleri kullanmak isteyen yeni ya da var olan müşteriler.
- Herhangi bir güncel LDAP iş mimarisiyle yerleşik ileti modelini doğrudan kullanmak veya ayarlamak isteyen yeni veya var olan müşteriler.

Desteklenen Ortamlar

Connection Endpoint Lookup örneğini çalıştırmadan önce, desteklenen bir işletim sistemi ve ilgili yazılıma sahip olduğunu doğrulayın.

IBM WebSphere MQ Connection Endpoint Lookup için örnek program, aşağıdaki yazılımları gerektirir:

- IBM WebSphere MQ V7.0ya da üstü

- Tivoli Directory Server V6.3 Client ya da üstü

Desteklenen işletim sistemleri:

1. Windows (XP/2003/2008)
2. Solaris (SPARC ve x86-64)
3. AIX
4. Linux
   - RHEL v4 ve Sistem püzerinde v5
   - SUSE v9 ve Sistem püzerinde v10
   - RHEL v4 ve v5 System x32 bit ve x64 bit
   - SUSE v9 ve v10 System x32 bit ve x64 bit
5. HP IA64.

Not: Örnek program, z/OS, i/5ve HP PARISC platformlarına uygun değildir.

Kurma ve Yapılandırma

Çıkış modülü ve bağlantı uç noktası şeması kuruluyor ve yapılandırılıyor.

Çıkış modülünün takılması

WebSphere MQkuruluşu sırasında, çıkış modülü `tools/samples/c/preconnexit/bin`altında kuruludur. 32 bit altyapılar için, çıkış modülünün kullanılabilmesi için `exit/<install name>/'a` kopyalanması gerekir. 64 bit altyapılar için, çıkış modülünün kullanılabilmesi için önce exit64/<installation name>/üzerine kopyalanması gerekir.

Bağlantı Uç Noktası şemasının kurulması

Çıkış, Bağlantı Uç Noktası şemasını (*ibm-amq.schema*) kullanır. Çıkış kullanılabilmesi için, şema dosyasının herhangi bir LDAP sunucusuna aktarılması gerekir. Şemayı içe aktardıktan sonra, özniteliklere ilişkin değerler eklenmelidir.

Burada, Bağlantı Uç Noktası şemasının içe aktarılmasına ilişkin bir örnek vardır. Örneğin, IBM Tivoli Directory Server (ITDS) kullanıldığını varsayar.

- Ensure that IBM Tivoli Directory Server is running, then copy or FTP the *ibm-amq.schema* file to the ITDS server.
- ITDS sunucusunda, şemayı ITDS deposuna kurmak için aşağıdaki komutu girin; burada LDAP kimliği ve LDAP parolası LDAP sunucusunun kök DN 'si ve paroladır:

  ldapadd -D "LDAP ID" -w "LDAP parolası" -f ibm-amq.schema
- Komut penceresinde, aşağıdaki komutu girin ya da doğrulama için şemaya göz atmak için bir üçüncü kişi aracı kullanın:

  ldapsearch objectclass=ibm-amqClientBağlantısı

Şema dosyasının içe aktarılmasına ilişkin ek bilgi için LDAP Sunucusu belgelerinize bakın.

Yapılandırma

A new section called **PreConnect** must be added to the client configuration file say *mqclient.ini* file. PreConnect kısmı aşağıdaki anahtar sözcükleri içerir:

Modül : API çıkış kodunu içeren modülün adı. Bu alan modülün tam yolunu içeriyorsa, WebSphere MQ kuruluşundaki diğer *exit* ya da *exit64* klasörünün arandığı bir dosya olarak kullanılır.

İşlev : İşlevsel giriş noktasının, PreConnect çıkış kodunu içeren kitaplığa ilişkin adı. İşlev tanımlaması MQ_PRECONNECT_EXIT prototipine uygun olur.

Veri : Kanal tanımlamalarını içeren LDAP havuzunun URI 'si.

Aşağıdaki kod parçası, *mqclient.ini* dosyası için gereken değişiklikleri gösteren bir örnektir.

```
PreConnect:
  Module=amqlcelp
  Function=PreConnectExit
  Data=ldap://myLDAPServer.com:389/cn=wmq,ou=ibm,ou=com
  Sequence=1
```

Çıkıştan ve Şemaya Genel Bakış

Sözdizimi ve kuyruk yöneticiyle bağlantı kurmak için kullanılan parametreler.

WebSphere MQ v7.5 , çıkış modülündeki bir giriş noktası için aşağıdaki sözdizimini tanımlar.

```
void MQENTRY MQ_PRECONNECT_EXIT ( PMQNXP  pExitParms
                                , PMQCHAR pQMgrName
                                , PPMQCNO ppConnectOpts
                                , PMQLONG pCompCode
                                , PMQLONG pReason)
```

MQCONN/X çağrısı yürütme sırasında, WebSphere MQ C İstemcisi işlev sözdiziminin somutlamasını içeren çıkış modülünü yükler. Daha sonra kanal tanımlamalarını almak için bir çıkış işlevini çağırır. Daha sonra, alınan kanal tanımlamaları kuyruk yöneticiyle bağlantı kurmak için kullanılır.

Parametreler

pExitParms

Tip: PMQNXP giriş/çıkış

PreConnection çıkış değiştirgesi yapısı. Yapı, çıkışa ilişkin çağırıcı tarafından ayrılır ve sürdürür.

```
struct tagMQNXP
{
  MQCHAR4    StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQLONG    ExitId;             /* Type of exit */
  MQLONG    ExitReason;         /* Reason for invoking exit */
  MQLONG    ExitResponse;       /* Response from exit */
  MQLONG    ExitResponse2;      /* Secondary response from exit */
  MQLONG    Feedback;           /* Feedback code (reserved) */
  MQLONG    ExitDataLength;        /* Exit data length */
  PMQCHAR      pExitDataPtr;           /* Exit data */
  MQPTR        pExitUserAreaPtr;       /* Exit user area */
  PMQCD *    ppMQCDArrayPtr;        /* Array of pointers to MQCDs */
  MQLONG    MQCDArrayCount;        /* Number of entries found */
  MQLONG    MaxMQCDVersion;        /* Maximum MQCD version */
};
```

pQMgrAdı

Tip: PMQCHAR giriş/çıkış

Kuyruk yöneticisinin adı. On input, this parameter is the filter string supplied to the MQCONN API call through the **QMgrName** parameter. Bu alan boş, açık ya da belirli genel arama karakterleri içerebilir. Alan, çıkışa göre değiştirilir. Çıkış MQXR_TERM ile çağrıldığında, parametre NULL (boş değerli) olur.

ppConnectSeçenekleri

Tip: ppConnectGiriş/çıkış

MQCONNX 'in işlemini denetleyen seçenekler. Bu, MQCONN API çağrısının işlemini denetleyen MQCNO bağlantı seçenekleri yapısına bir işarettir. Çıkış MQXR_TERM ile çağrıldığında, parametre NULL (boş değerli) olur. MQI istemcisi, çıkışa her zaman, uygulama tarafından sağlanmamış olsa da, çıkışa bir MQCNO yapısı sağlar. Bir uygulama MQCNO yapısı sağlıyorsa, istemci bunu değiştirdiği yerden çıkışa geçirmek için bir yineleme yapar. İstemci MQCNO ' nun sahipliğini korur. MQCNO ile gönderme yapılan bir MQCD, dizi aracılığıyla sağlanan bağlantı tanımlarına göre öncelik kazanır. İstemci, kuyruk yöneticisine bağlanmak için MQCNO yapısını kullanıyor ve diğerleri yok sayılıyor.

pCompKodu

Tip: PMQXX_ENCODE_CASE_ONE long giriş/çıkış

Tamamlanma kodu. Çıkışların tamamlanma kodunu alan bir MQUZE işaretçisi. Bu değer aşağıdaki değerlerden biri olmalıdır:

MQCC_OK-Başarılı tamamlanma

MQCC_UYARI-Uyarı (kısmi tamamlama)

MQCC_FAILED-Çağrı başarısız oldu

pReason

Tip: PMQXX_ENCODE_CASE_ONE long giriş/çıkış

Neden niteleyici pCompkodu. Çıkış neden kodunu alan bir MQUZE işaretçisi. Tamamlanma kodu MQCC_OK ise, tek geçerli değer şöyledir:

MQRC_NONE-(0, x '000') Raporlamak için bir neden yok.

Tamamlanma kodu MQCC_FAILED ya da MQCC_UYARI ise, çıkış işlevi neden kodu alanını geçerli bir MQRC_ * değerine ayarlayabiliyor.

MQ LDAP Bağlam Bilgileri

Çıkış, bağlam bilgileri için aşağıdaki veri yapısını kullanır.

MQNLDAPCTX

MQNLDAPCTX yapısı aşağıdaki C prototipine sahiptir.

```
typedef struct tagMQNLDAPCTX MQNLDAPCTX;
typedef MQNLDAPCTX MQPOINTER PMQNLDAPCTX;

struct tagMQNLDAPCTX
{
  MQCHAR4    StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  LDAP *    objectDirectory;    /* LDAP Instance */
  MQLONG       ldapVersion;        /* Which LDAP version to use? */
  MQLONG       port;            /* Port number for LDAP server*/
  MQLONG       sizeLimit;         /* Size limit */
  MQBOOL    ssl;            /* SSL enabled? */
  MQCHAR *    host;            /* Hostname of LDAP server */
  MQCHAR *    password;        /* Password of LDAP server */
  MQCHAR *    searchFilter;        /* LDAP search filter */
  MQCHAR *    baseDN;           /* Base Distinguished Name */
  MQCHAR *    charSet;        /* Character set */
};
```

Bağlantı uç noktası arama çıkışını oluşturmak için örnek kod

Kaynağı Windows ve dağıtılmış altyapılarda derlemek için kullanılan kod parçacıklar.

Kaynak derleniyor

Kaynağı herhangi bir LDAP istemci kitaplıkla derleyebilirsiniz; örneğin, IBM Tivoli Directory Server 6.3 Client kitaplıkları. Bu belgeler, Tivoli Directory Server 6.3 istemci kitaplıklarını kullandığınızı varsayar.

Not: Bağlantı öncesi çıkış kitaplığı, aşağıdaki LDAP sunucularıyla sınanmıştır:

- IBM Tivoli Directory Server V6.3
- Novell eDirectory V8.2

Aşağıdaki kod parçacıkları, Pencereler' ta ve diğer dağıtılmış altyapılarda çıkışlar nasıl derleneceğini açıklar:

Compiling the exit on the Pencereler platform

You can use the following snippet for compiling the exit source on Pencereler:

```
CC=cl.exe
LL=link.exe
CCARGS=/c /I. /DWIN32 /W3 /DNDEBUG /EHsc /D_CRT_SECURE_NO_DEPRECATE /Zl

# The libraries to include
LDLIBS=Ws2_32.lib Advapi32.lib libibmldapstatic.lib libibmldapdbgstatic.lib  \
kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib advapi32.lib \
shell32.lib ole32.lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib msvcrt.lib
```

```
OBJS=amqlcel0.obj

all: amqlcelp.dll

amqlcelp.dll: $(OBJS)
        $(LL) /OUT:amqlcelp.dll /INCREMENTAL /NOLOGO /DLL /SUBSYSTEM:WINDOWS /MACHINE: X86 /
DEF:amqlcelp.def $(OBJS) $(LDLIBS) /NODEFAULTLIB:msvcrt.lib

# The exit source
amqlcel0.obj: amqlcel0.c
              $(CC) $(CCARGS) $*.c
```

Not: Microsoft Visual Studio 2003 derleyicisiyle derlenen IBM Tivoli Directory Server 6.3 Client kitaplıklarını kullanıyorsanız, IBM Tivoli Directory Server 6.3 Client kitaplıklarını Microsoft Visual Studio 2005 ya da üstü derleyici ile derlerken uyarılar alabilirsiniz.

Diğer dağıtılmış platformlardaki çıkışa derleniyor

Çıkış kaynağını diğer dağıtımlı altyapılarda derlemek için aşağıdaki parçacığı kullanabilirsiniz; örneğin, Linux. Bazı derleyici seçenekleri, diğer dağıtılmış altyapılarda farklı olabilir.

```
#Make file to build exit
CC=gcc

MQML=/opt/mqm/lib
MQMI=/opt/mqm/inc
TDSI=/opt/ibm/ldap/V6.3/include
XFLAG=-m32

TDSL=/opt/ibm/ldap/V6.3/lib
```

IBM Tivoli Directory Server hem statik, hem de dinamik bağlantı kitaplıklarını gönderir, ancak kitaplıkların yalnızca bir biçimi kullanılabilir. Bu komut dosyası, statik kitaplıkları kullandığınızı varsayar.

```
#Use static libraries.
LDLIBS=-L$(TDSL) -libibmldapstatic

CFLAGS=-I. -I$(MQMI) -I$(TDSI)

all:amqlcepl

amqlcepl: amqlcel0.c
        $(CC) -o cepl amqlcel0.c  -shared -fPIC $(XFLAG) $(CFLAGS) $(LDLIBS)
```

Modül Çağırımı Çık

PreConnect çıkış modülü, üç farklı neden koduyla çağrılabilir. Bu bölümde, daha fazla derinlikte her bir çıkış nedeni anlatılır.

MQXR_INIT

Çıkış, bir LDAP sunucusuyla bağlantı kurmak ve kullanıma hazırlamak için MQXR_INIT neden koduyla çağrılır.

MQXR_INIT çağrısından önce, MQNXP yapısının *pExitDataPtr* alanı, *mqclient.ini* dosyası (örn. LDAP) içindeki PreConnect stanza içindeki Veri özniteliği ile doldurulcaktı.

LDAP URL adresi, arama için en az protokol, anasistem adı, kapı numarası ve temel DN ' den oluşur. Çıkış, *pExitDataPtr* alanında bulunan LDAP URL 'sini ayrıştırır ve bir MQNLDAPCTX LDAP Lookup Context yapısı ayırır ve buna uygun şekilde doldurur. Bu yapının adresi, *pExitUserAreaPtr* alanında saklanır. LDAP URL sonuçlarını doğru olarak ayrıştıramazsanız, MQCC_FAILED hatası oluştu.

Bu noktada, çıkış, MQNLDAPCTX değiştirgelerini kullanarak LDAP sunucusuna bağlanır ve bağlanıyor. Sonuçta elde edilen LDAP API tanıtıcıları da bu yapı içinde saklanır.

MQXR_PRECONNECT

Çıkış modülü, bir LDAP sunucusundan kanal tanımlamalarını almak için MQXR_PRECONNECT neden koduyla çağrılır.

Çıkış, belirtilen süzgeçle eşleşen kanal tanımlamaları için LDAP sunucusunda arama yapar. *QMgrName* parametresi belirli bir kuyruk yöneticisi adı içeriyorsa, arama, *ibm-amqQueueManagerName* LDAP özniteliği değeri verilen kuyruk yöneticisi adına sahip olan tüm kanal tanımlamalarını döndürür.

QMgrName değiştirgesi '*' ya da ' ise ' (boşluk), arama, *ibm-amqIsClientDefault* Connection uç noktası özniteliği true olarak ayarlanana tüm kanal tanımlamalarını döndürür.

Başarılı bir aramadan sonra, çıkış bir ya da bir MQCD tanımı dizisi hazırlar ve çağıranı geri döndürür.

MQXR_TERM
Çıkış temizlenmek olduğunda, çıkış bu neden koduyla çağrılır. Bu çıkış sırasında, çıkış LDAP sunucusundan kesilir ve çıkışta ayrılan ve bakımı yapılan tüm belleği serbest bırakır. Bu işlem, *MQNLDAPCTX* yapısını, gösterge dizisini ve gönderme yaptığı her MQCD ' yi içerir. Diğer alanlar varsayılan değerlere ayarlanır. *pQMgrName* ve *ppConnectOpts* çıkış değiştirgeleri *MQXR_TERM* sırasında kullanılmamış ve NULL (boş) olabilir.

LDAP şemaları
İstemci bağlantı verileri, LDAP (LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizini adı verilen bir genel havuzda saklanır. Bir WebSphere MQ istemcisi, bağlantı tanımlarını almak için bir LDAP dizini kullanır. LDAP dizini içindeki WebSphere MQ istemci bağlantısı tanımlamalarının yapısı LDAP şeması olarak bilinir. LDAP şeması, öznitelik tipi tanımlamalarının, nesne sınıfı tanımlamalarının ve sunucunun, bir süzgeç ya da öznitelik değeri değerlendirmesinin bir girişin öznitelikleriyle eşleşip eşleşmediğini ve işlemlerin izin, ekleme ve değiştirme işlemlerine izin verilip verilmeyeceğini belirlemek için kullandığı diğer bilgiler toplamlıdır.

Verileri LDAP dizininde saklama

İstemci bağlantı tanımları, bağlantı noktası olarak bilinen dizin ağacı içindeki belirli bir dalın altında bulunur. Bir LDAP dizinindeki diğer tüm düğümler gibi, bağlantı noktasının ilişkili bir Ayırt Edici Adı (DN) vardır. Bu düğümü, dizin üzerinde yaptığınız tüm sorguların başlangıç noktası olarak kullanabilirsiniz. İstemci bağlantı tanımlarının bir alt kümesini döndürmek için LDAP dizini sorgulanırken süzgeç uygulamayı kullanın. Dizin ağacının diğer bölümlerinde verilen izinlere dayalı olarak alt ağaçlara erişimi kısıtlayabilirsiniz; örneğin, kullanıcılara, bölümlere ya da gruplara.

Kendi özniteliklerinizi ve sınıflarınızı tanımlama
LDAP şemasını değiştirerek istemci kanalı tanımlamasını saklayın. Tüm LDAP veri tanımlamaları nesne ve öznitelik gerektirir. Nesneler ve öznitelikler, nesneyi ya da özniteliği benzersiz bir şekilde tanımlayan bir nesne tanıtıcısı (OID) numarasıyla tanımlanır. Bir LDAP şemasının içindeki tüm sınıflar, doğrudan ya da dolaylı olarak üst nesneden devralır. İstemci kanalı tanımlaması nesnesi, üst nesnenin özniteliklerini içerir. Tüm LDAP veri tanımlamaları nesne ve öznitelik gerektirir:

- Nesne tanımlamaları, LDAP özniteliklerinin derlemleridir.
- Öznitelikler LDAP veri tipleridir.

Her bir özniteliğin açıklaması ve bu özelliklerin normal WebSphere MQ özellikleriyle nasıl eşleneceği LDAP öznitelikleri içinde açıklanmıştır.

LDAP öznitelikleri
Tanımlanan LDAP öznitelikleri, WebSphere MQ ' ya özgülenir ve doğrudan istemci bağlantısı özellikleriyle eşlenir.

WebSphere MQ Client Channel Directory String Öznitelikleri
The character string attributes with their mapping to WebSphere MQ properties are listed in the following table. Öznitelikler, directoryString (UTF-8 kodlamalı Unicode) değerlerini, bir alt küme olarak IA5/ASCII içeren bir değişken bayt kodlama sistemi olan değerleri tutabilir. Sözdizimi, nesne tanıtıcısı numarası (OID) ile belirtilir.

Çizelge 21. WebSphere MQ istemcisi kanal dizini dizgi öznitelikleri		
LDAP Özniteliği	**Tanım**	**WebSphere MQ Özelliği**
CN	Kanal adından ve tanımlayıcı kuyruk yöneticisi adına sahip ortak ad.	
ibm-amqChannelAdı	Kanal tanımlamasının adı.	Kanal
ibm-amqConnectionAdı	İletişim bağlantısı tanıtıcısı.	ADı
ibm-amqDescription	Kanal açıklaması.	TASARIMLA
ibm-amqLocalAdresi	Kanala ilişkin yerel iletişim adresi.	KAPSAYıCı
ibm-amqModeAdı	A bThe LU 6.2 kip adıdır.	MODENAME
ibm-amqPassword	Kullanılabilecek parola.	Parola
ibm-amqQueueManagerName	Bir WebSphere MQ istemcisi uygulamasının bağlantı isteyebileceği kuyruk yöneticisi ya da kuyruk yöneticisi grubunun adı.	QMNAME
ibm-amqSecurityExitUserVerileri	Güvenlik çıkışa geçirilen kullanıcı verileri.	SCYDATA
ibm-amqSecurityExitName	Kanal güvenliği çıkışıyla çalıştırılacak çıkış programının adı.	SCYEXIT
ibm-amqSslCipherSpec	SSL bağlantısı için tek bir CipherSpec .	SSLCIPH
ibm-amqSslPeerName	Bir WebSphere MQ kanalının diğer ucundaki eşdüzey kuyruk yöneticisinden ya da istemciden alınan sertifikana ilişkin ayırt edici adı (DN) denetler.	SSLPEER
ibm-amqTransactionProgramName	Hareket programı adı.	TADı
ibm-amqUserTanıtıcısı	Uzak MCA ile güvenli bir SNA oturumu başlatma girişiminde bulunulduğunda MCA tarafından kullanılacak kullanıcı kimliği.	USERID

WebSphere MQ istemcisi bağlantı tamsayı öznitelikleri

Önceden tanımlanmış değerlere sahip öznitelikler (örneğin, bir sıralı tip) standart tamsayılar olarak depolanır. Bu değerler LDAP dizininde tamsayı değerleri olarak saklanır ve ilişkili sabit ad kullanılarak değil.

Çizelge 22. WebSphere MQ istemcisi kanal dizini tamsayı öznitelikleri		
LDAP Özniteliği	**Tanım**	**WebSphere MQ Özelliği**
ibm-amqConnectionAffinity	Aynı kuyruk yöneticisi adı üzerinden birden çok kez bağlanan istemci uygulamalarının aynı istemci kanalını kullanıp kullanmayacağını belirler.	BENZERLIK
ibm-amqClientChannelWeight	İstemci bağlantı kanalı tanımlamasının kullanıldığı etki alanı ağırlıklandırma.	CLNTWGHT
ibm-amqHeartBeatInterval	İletim kuyruğunda ileti olmadığında, gönderen MCA ' dan geçirilecek sağlıklı işletim bildirimi akışları arasındaki yaklaşık süre.	HBNT
ibm-amqKeepAliveInterval	Bir kanala ilişkin zaman aşımı değeri.	KAINT

Çizelge 22. WebSphere MQ istemcisi kanal dizini tamsayı öznitelikleri (devamı var)		
LDAP Özniteliği	**Tanım**	**WebSphere MQ Özelliği**
ibm-amqMaximumMessageLength	Kanalda iletilebilecek ileti uzunluğu üst sınırı.	MAXMSGL
ibm-amqSharingKonuşmaları	Her bir TCP/IP kanalı yönetim ortamını paylaşan etkileşim sayısı üst sınırı.	SHARECNV
ibm-amqTransportTipi	Kullanılacak iletim tipi.	TRPTYPE

WebSphere MQ istemcisi kanal boole özniteliği

Bu Boole özniteliği herhangi bir WebSphere MQ özelliğiyle eşlenmez. Bu özniteliğin sözdizimi, bir boole değeri gösterir.

Çizelge 23. WebSphere MQ istemcisi kanal boole özniteliği	
LDAP Özniteliği	**Tanım**
ibm-amqIsClientDefault	Bu Boole özniteliği, ibm-amqQueueManagerName özniteliği tanımlı olmayan girişlerin aranması sorununu çözmek için tanımlanır.

WebSphere MQ istemcisi kanal listesi öznitelikleri

WebSphere MQ özellikleri, LDAP dizini içinde tek değerli, virgülle ayrılmış liste özniteliği olarak depolanır. Öznitelikler, diğer dizin dizesi öznitelikleriyle aynı şekilde tanımlanır. Liste öznitelikleri, WebSphere MQ özelliklerine eşlemeleriyle birlikte aşağıdaki tabloda açıklanmıştır.

Çizelge 24. WebSphere MQ istemcisi kanal listesi öznitelikleri		
LDAP Özniteliği	**Tanım**	**WebSphere MQ Özelliği**
ibm-amqHeaderSıkıştırması	Kanal tarafından desteklenen üstbilgi veri sıkıştırma tekniklerini içeren bir listedir.	KARMAşıK
ibm-amqMessageSıkıştırması	Kanal tarafından desteklenen ileti veri sıkıştırma tekniklerinin listesi.	MSG
ibm-amqSendExitUserVerileri	Gönderme çıkışa geçirilen kullanıcı verileri.	SENDDATA
ibm-amqSendExitUserAdı	Kanal gönderme çıkışıyla çalıştırılacak çıkış programının adı.	SENDEXIT
ibm-amqReceiveExitUserVerileri	Alma çıkışa geçirilen kullanıcı verileri.	RVDATA
ibm-amqReceiveExitName	Kanal tarafından çalıştırılacak kullanıcı çıkış programının adı, kullanıcı çıkışı alır.	RCVEXIT

Ortak Ad

Ortak ad (CN), kanal adından ve tanımlayıcı kuyruk yöneticisi adından oluşur.

Bu önceden var olan bir öznitedir.

CN ' nin biçimi şöyledir:

```
CN=CHANNEL_NAME(DEFINING_Q_MGR_NAME)
```

Örneğin:

```
CN=TC1(QM_T1)
```

Bu öznitelik için yalnızca bir değer belirtebilirsiniz.

Bu öznitelik bir dizgi öznitelidir ve değerler büyük/küçük harfe duyarlı değildir. Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, alt şemada kullanılan, bir alt dizgiyle (örneğin, CN=jim * bir özniteliğe sahip CN=jim * gibi) bir alt şemada özniteliğin davranışını belirleyen ve bir ya da daha fazla joker karakter içeren eşleştirme kuralıdır.

ibm-amqChannelAdı
Bu öznitelik, kanal tanımlamasının adını belirtir.

Bu özniteliğin, büyük/küçük harf duyarlı olmayan en fazla 20 karakteri içeren tek bir dizgi değeri vardır. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, alt şemada kullanılan, bir alt dizgiyi kullanan ve bir ya da daha fazla joker karakter içeren bir arama süzgecindeki özniteliğin davranışını belirleyen eşleşen bir kuraldır.

ibm-amqDescription
Bu LDAP özniteliği kanal tanımlamasını sağlar.

Bu özniteliğin, büyük/küçük harf duyarlı olmayan tek bir dizgi değeri en çok 64 byte olmalıdır. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqConnectionAdı
Bu LDAP özniteliği, iletişim bağlantı tanıtıcısıdır. Bu kanal tarafından kullanılacak iletişim bağlantılarını belirler.

Bu özniteliğin en çok 264 karakteri olan tek bir dizgi değeri vardır; bu, büyük/küçük harfe duyarlı değildir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqLocalAdresi
Bu öznitelik, kanala ilişkin yerel iletişim adresini belirtir.

Bu özniteliğin en çok 48 karakterden oluşan tek bir dizgi değeri vardır; bu, büyük/küçük harfe duyarlı değildir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqModeAdı
Bu öznitelik, LU 6.2 bağlantılarıyla birlikte kullanılmak içindir. Bir iletişim oturumu ayırma işlemi gerçekleştirildiğinde bağlantının oturum özellikleri için ek tanım sağlar.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, tam olarak 8 karakterden oluşan tek bir dizgi değerine sahiptir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqPassword
Bu LDAP özniteliği, uzak MCA ile güvenli bir LU 6.2 oturumu başlatma girişimi sırasında MCA tarafından kullanılabilecek bir parola belirtir.

Bu özniteliğin en çok 12 hanesi olan tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir.

ibm-amqQueueManagerName
Bu öznitelik, bir WebSphere MQ istemcisi uygulamasının bağlantı isteyebileceği kuyruk yöneticisi ya da kuyruk yöneticisi grubunun adını belirtir.

Bu özniteliğin en çok 48 karakterden oluşan tek bir dizgi değeri vardır; bu, büyük/küçük harfe duyarlı değildir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqSecurityExitUserVerileri
Bu LDAP özniteliği, güvenlik çıkışa geçirilen kullanıcı verilerini belirtir.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, en çok 999 karakterden oluşan tek bir dizgi değerine sahiptir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqSecurityExitName
Bu LDAP özniteliği, kanal güvenliği çıkışıyla çalıştırılacak çıkış programının adını belirtir.

Kanal güvenlik çıkışı yoksa, boş bırakın.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, en çok 999 karakterden oluşan tek bir dizgi değerine sahiptir. Bu öznitelik, çıkış öncesi bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqSslCipherSpec
Bu LDAP özniteliği, SSL bağlantısı için tek bir CipherSpec belirtir.

Bu özniteliğin, büyük/küçük harf duyarlı olmayan tek bir dizgi değeri en çok 32 karakter olmalıdır. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqSslPeerName
Bu LDAP özniteliği, bir WebSphere MQ kanalının diğer ucundaki eşdüzey kuyruk yöneticisinden ya da istemciden alınan sertifikana ilişkin ayırt edici adı (DN) denetlemek için kullanılır.

Bu LDAP özniteliğinin, büyük/küçük harf duyarlı olmayan tek bir dizgi değeri en fazla 1024 byte olmalıdır. Bu önceden var olan bir şey değil.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqTransactionProgramName
Bu LDAP özniteliği, hareket programı adını belirtir. Bu, LU 6.2 bağlantılarıyla kullanılmak içindir.

Bu özniteliğin, büyük/küçük harf duyarlı olmayan, en çok 64 karakterden oluşan tek bir dizgi değeri vardır. Bu önceden var olan bir şey değil.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqUserTanıtıcısı
Bu LDAP özniteliği, uzak MCA ile güvenli bir SNA oturumu başlatma girişiminde bulunulduğunda MCA tarafından kullanılacak kullanıcı kimliğini belirtir.

Bu özniteliğin tek bir dizgi değeri tam 12 karakter (büyük/küçük harfe duyarlı değildir). Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

ibm-amqConnectionAffinity
Bu LDAP özniteliği, aynı kuyruk yöneticisi adını kullanarak birden çok kez bağlantı yapan istemci uygulamalarının aynı istemci kanalını kullanıp kullanmadığını belirler.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir.

ibm-amqClientChannelWeight
Bu LDAP özniteliği, hangi istemci bağlantısı kanal tanımlamasının kullanıldığını etkileyen bir ağırlıklandırma belirtir.

İstemci kanalı ağırlıklandırma özniteliği, birden çok uygun tanımlama kullanılabilir olduğunda istemci kanalı tanımlarının seçilmesini önlemek için kullanılır.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir.

ibm-amqHeartBeatInterval
Bu LDAP özniteliği, iletim kuyruğunda herhangi bir ileti olmadığında, gönderen MCA ' dan geçirilecek sağlıklı işletim bildirimi akışları arasındaki yaklaşık süreyi belirtir.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir. Varsayılan değer 1 'dir. Varsayılan değer, yürürlükteki MQSERVER ortam değişkeni işleminde ayarlıdır.

ibm-amqKeepAliveInterval
Bu LDAP özniteliği, bir kanala ilişkin bir zaman aşımı değeri belirtmek için kullanılır.

Bu özniteliğin değeri, kanala ilişkin canlı tutma (Keepalive) zamanlaması belirterek, iletişim yığınına geçirilir. Bu seçeneği, her kanal için farklı bir canlı tutma değeri belirtmek için kullanabilirsiniz.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir.

ibm-amqMaximumMessageLength
Bu LDAP özniteliği, kanalda iletilebilecek bir ileti uzunluğu üst sınırını belirtir.

Bu özniteliğin varsayılan değeri, yürürlükteki MQSERVER ortam değişkeni işlemine göre 104857600 'tür. Bu özniteliğin tek bir tamsayı değeri var ve bu öznitelik önceden var olan bir öznitelik değil.

ibm-amqSharingKonuşmaları
Bu LDAP özniteliği, her bir TCP/IP kanalı yönetim ortamını paylaşan etkileşim sayısı üst sınırını belirtir.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu öznitelik, önceden var olan bir öznitelik değil.

ibm-amqTransportTipi
Bu LDAP özniteliği, kullanılacak iletim tipini belirtir.

Bu özniteliğin tek bir tamsayı değeri vardır. Bu, önceden var olan bir öznitelik değildir.

ibm-amqIsClientDefault
Bu Boole özniteliği, ibm-amqQueueManagerName özniteliğinin tanımlanmadığı arama girdilerinin sorununu çözer.

Önyükleme öncesi çıkış birimleri genellikle, arama ölçütü olarak, LDAP sunucularında ibm-amqQueueManagerName özniteliğinin değerini içeren LDAP sunucularını arar. Bu tür bir sorgu, ibm-amqQueueManagerName öznitelik değerinin, MQCONN/X çağrısında belirlenen kuyruk yöneticisinin adıyla eşleştiği tüm girişleri döndürür. Ancak, istemci kanal tanımlama çizelgeleri (CCDT) kullanılırken, bir MQCONN/X çağrısında kuyruk yöneticisi adını boş olarak ayarlayabilir ya da adın önekini yıldız (*) ile önleyebilirsiniz. Kuyruk yöneticisinin adı boş bırakılırsa, istemci varsayılan kuyruk yöneticisine bağlanır. Adın başına bir yıldız işareti (*) eklenirse, istemci kuyruk yöneticisini bağlar.

Benzer şekilde, bir girişteki ibm-amqQueueManagerName özniteliği tanımsız bırakılabilir. Bu durumda, bu uç nokta bilgilerini kullanan istemcinin herhangi bir kuyruk yöneticisine bağlanabilmesi beklenir. Örneğin, bir girdi aşağıdaki satırları içerir:

```
ibm-amqChannelName = "CHANNEL1"
ibm-amqConnectionName = myhost(1414)
```

Bu örnekte, istemci, `myhost`üzerinde çalışan belirtilen kuyruk yöneticisine bağlanmayı dener.

Ancak LDAP Sunucularında, tanımlı olmayan bir öznitelik değerinde arama yapılmaz. Örneğin, bir giriş, ibm-amqQueueManagerNamedışındaki bağlantı bilgilerini içeriyorsa, arama sonuçları bu girişi içermeyecekti. Bu sorunu aşmak için ibm-amqIsClientDefaultöğesini ayarlayabilirsiniz. Bu bir Boole öznitelidir ve tanımlanmamışsa, FALSE değerinin olduğu varsayılır.

Ibm-amqQueueManagerName ' in tanımlanmadığı ve aramanın bir parçası olması beklenen girişler için, ibm-amqIsClientDefault değerini TRUE olarak ayarlayın. MQCONN/X çağrısında kuyruk yöneticisi adı olarak boş ya da yıldız işareti (*) belirtildiğinde, ön bağlanma çıkışı, ibm-amqIsClientDefault öznitelik değerinin TRUE olarak ayarlandığı tüm girişler için LDAP sunucusunu arar.

Not: Ibm-amqIsClientDefault TRUE olarak ayarlandıysa, ibm-amqQueueManagerName özniteliğini ayarlamaz ya da tanımlamayın.

ibm-amqHeaderSıkıştırma
Bu LDAP özniteliği, kanal tarafından desteklenen üstbilgi veri sıkıştırma tekniklerinin bir listesidir.

Bu özniteliğin büyüklük üst sınırı 48 karakterdir. Bu, önceden var olan bir öznitelik değildir.

Bu öznitelik için yalnızca bir değer belirtebilirsiniz.

Bu liste özniteliği, virgülle ayrılmış bir biçim kullanılarak dizin dizgileri olarak belirtilir. Örneğin, **ibm-amqHeaderCompression** için belirtilen değerler `0` , bu değer `NONE`olarak eşlenir. İzin verilen üst sınırı aşan değerler, istemci tarafından yok sayılacaktır. Örneğin, ibm-amqHeaderCompression, listede en çok 2 tamsayı içerir.

ibm-amqMessageSıkıştırma
Bu LDAP özniteliği, kanal tarafından desteklenen ileti veri sıkıştırma tekniklerinin bir listesidir.

Bu özniteliğin büyüklük üst sınırı 48 karakterdir. Bu, önceden var olan bir öznitelik değildir.

Bu öznitelik birden çok değeri desteklemiyor.

Bu liste özniteliği, virgülle ayrılmış bir biçim kullanılarak dizin dizgileri olarak belirtilir. Örneğin, bu öznitelik için belirtilen değer 1,2,4 'tür ve bu, temeldeki sıkıştırma sırası RLE, ZLIBFAST ve ZLIBSTHH ile eşlenir.

İzin verilen üst sınır değerini aşan değerler istemci tarafından yoksayılır. Örneğin, ibm-amqMessageSıkıştırması, listede en çok 16 tamsayı içerir.

ibm-amqSendExitUserVerileri
Bu LDAP özniteliği, gönderme çıkışa geçirilen kullanıcı verilerini belirtir.

Bu LDAP özniteliğinin, büyük ve küçük harfe duyarlı olmayan tek bir dizgi değeri en fazla 999 karakter içerir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

Not: ibm-amqSendExitName ve **ibm-amqSendExitUserData** ' in çift olarak eşitlenmesi gerekir. Kullanıcı verileri, çıkış adıyla uyumlulaştırılmalıdır. Bu nedenle, biri belirtilirse, diğeri veri içermese de, diğer yandan da simetrik olarak belirtilmiş olmalıdır.

ibm-amqSendExitName
Bu LDAP özniteliği, kanal gönderme çıkışıyla çalıştırılacak çıkış programının adını belirtir.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, en çok 999 karakterden oluşan tek bir dizgi değerine sahiptir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

Not: ibm-amqSendExitName ve **ibm-amqSendExitUserData** , çiftler halinde eşitlenmiş olmalıdır. Kullanıcı verileri, çıkış adıyla eşitlenmiş olmalıdır. Bu nedenle, biri belirtilirse, diğeri veri içermese de simetrik olarak belirtilmiş olmalıdır.

ibm-amqReceiveExitUserVerileri
Bu LDAP özniteliği, alma çıkışa geçirilen kullanıcı verilerini belirtir.

Bir dizi alma çıkışı çalıştırabilirsiniz. Bir dizi çıkışa ilişkin kullanıcı verileri dizgisi virgülle, boşluklarla ya da her ikisiyle birbirinden ayrılır.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, en çok 999 karakterden oluşan tek bir dizgi değerine sahiptir. Bu, önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

Not: `ibm-amqReceiveExitName` ve **`ibm-amqReceiveExitUserData`** , çiftler halinde eşitlenmiş olmalıdır. Kullanıcı verileri, çıkış adıyla eşitlenmiş olmalıdır. Bu nedenle, biri belirtilirse, diğeri veri içermese de simetrik olarak belirtilmiş olmalıdır.

ibm-amqReceiveExitName
Bu LDAP özniteliği, kanal tarafından çalıştırılacak kullanıcı çıkış programının adını belirtir.

Bu öznitelik, art arda çalıştırılacak programların adlarının bir listesidir. Herhangi bir kanal alma kullanıcı çıkışı yürürlükte değilse, boş bırakın.

Bu öznitelik, büyük/küçük harf duyarlı olmayan, en çok 999 karakterden oluşan tek bir dizgi değerine sahiptir. Önceden var olan bir öznitelik değildir.

Alt dizgi eşleştirme yoksayıldı. Alt dizgi eşleştirme, bir arama süzgecindeki özniteliğin davranışını belirleyen, alt şemada kullanılan bir eşleşen kuraldır.

Not: `ibm-amqReceiveExitName` ve **`ibm-amqReceiveExitUserData`** , çiftler halinde eşitlenmiş olmalıdır. Kullanıcı verileri, çıkış adıyla eşitlenmiş olmalıdır. Bu nedenle, biri belirtilirse, diğeri veri içermese de, simetrik olarak da belirtilmiş olmalıdır.

Kuyruğa alma uygulaması yazılıyor

Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

Uygulama yazma hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

İlgili kavramlar

“Uygulama geliştirme kavramları” sayfa 7
IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“IBM WebSphere MQ uygulamalarını tasarlama” sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

“Örnek WebSphere MQ programları” sayfa 92
Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

“İstemci uygulamaları yazılıyor” sayfa 335
What you need to know to write client applications on WebSphere MQ.

“ WebSphere MQ' da Web hizmetlerini kullanma” sayfa 907
SOAP için IBM WebSphere MQ iletimi ya da HTTP için IBM WebSphere MQ köprüsü kullanılarak Web hizmetleri için IBM WebSphere MQ uygulamaları geliştirebilirsiniz.

“IBM WebSphere MQ uygulaması oluşturulması” sayfa 409

Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

"Program hatalarının işlenmesi" sayfa 526
Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

Message Queue Interface-Genel Bakış

Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

İleti Kuyruğu Arabirimi aşağıdaki öğelerden oluşur:

- Programların kuyruk yöneticisine ve tesislerine erişebileceği*Çağrılar*
- queue, veri aktarmak ve kuyruk yöneticisinden veri almak için kullanılan*Yapılar*
- Veri aktarma ve kuyruk yöneticisinden veri almak için*temel veri tipleri*

WebSphere MQ for Windows and WebSphere MQ on UNIX and Linux systems also supply:

- Calls through which WebSphere MQ for Windows and WebSphere MQ on UNIX and Linux systems programs can commit and back out changes.
- *İçerme dosyaları* bu altyapılarda sağlanan değişmezlerin değerlerini tanımlar.
- Uygulamalarınızı bağlamak için*Kitaplık dosyaları* .
- Bu altyapılarda MQI ' yi nasıl kullanacağını gösteren örnek programlar grubu. Bu örneklerle ilgili daha fazla bilgi için bkz. "Dağıtılmış platformlar için örnek programlar" sayfa 92.
- Dış hareket yöneticilerine bağ tanımları için örnek kaynak ve yürütülebilir kod.

MQI hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- "MQI çağrıları" sayfa 188
- "Eşitleme noktası aramaları" sayfa 188
- "Veri dönüştürme, veri tipleri, veri tanımları ve yapılar" sayfa 189
- "IBM WebSphere MQ kod parçası programları ve kitaplık dosyaları" sayfa 189
- "Tüm çağrılar için ortak olan parametreler" sayfa 194
- "Arabelleklerin belirtilmesi" sayfa 195
- "UNIX and Linux sinyal işleme" sayfa 195

İlgili kavramlar

"Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme" sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

"Nesnelerin açılması ve kapatılması" sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

"İletileri Kuyruğa Koyma" sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

"Kuyruktan Ileti Alınması" sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

"Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme" sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

"İş birimlerinin kesinleştirilmesi ve yedeklenmesi" sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

"Starting IBM WebSphere MQ applications using triggers" sayfa 313

Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

"MQI ve kümelerle çalışma" sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

MQI çağrıları

MQI ' daki çağrıları öğrenmek için bu bilgileri kullanın.

MQI ' deki çağrılar aşağıdaki gibi gruplanabilir:

MQCONN, MQCONNX ve MQDISC
Bu çağrıları kullanarak, (seçeneklerle ya da seçeneksiz) bir programı, kuyruk yöneticisinden bir program bağlantısını kesin (bir program ile). z/OSiçin CICS programları yazarsanız, bu çağrıları kullanmanız gerekmez. Ancak, uygulamanızı diğer altyapılara kapılamak istiyorsanız, bunları kullanmanız önerilir.

MQOPEN ve MQCLOSE
Bir nesneyi (kuyruk gibi) açmak ve kapatmak için bu çağrıları kullanın.

MQPUT ve MQPUT1
Bir ileti kuyruğuna ileti koymak için bu çağrıları kullanın.

MQGet
Bir kuyruktaki iletilere göz atmak ya da kuyruktan iletileri kaldırmak için bu çağrıyı kullanın.

MQSUB, MQSUBRQ
Bir konuya abonelik kaydetmek ve abonelikle eşleşen yayınları istemek için bu çağrıları kullanın.

MQINQ
Bir nesnenin özniteliklerine ilişkin bilgi edinmek için bu aramayı kullanın.

MQSET
Bir kuyruğun özniteliklerinin bazılarını ayarlamak için bu çağrıyı kullanın. Diğer nesne türlerinin özniteliklerini ayarlayamazsınız.

MQBEGIN, MQCMIT ve MQBACK
WebSphere MQ bir iş biriminin eşgüdümcüsü olduğunda bu çağrıları kullanın. MQBEGIN, iş birimini başlatır. MQCMIT ve MQBACK, iş birimi sırasında yapılan güncellemeleri kesinleştirerek ya da geri döndürerek iş birimini sona erdirir. Yerel başlatma kesinleştirme denetimi, kesinleştirme ve geri alma komutları kullanılır.

MQCRTMH, MQBUFMH, MQMHBUF, MQDLTMH
Bir ileti tanıtıcısı yaratmak, ileti tanıtıcısını bir arabelleğe ya da arabelleğe bir ileti tanıtıcısı oluşturmak ve bir ileti tanıtıcısını silmek için bu çağrıları kullanın.

MQSETMP, MQINQMP, MQDLTMP
İleti tanıtıcısı üzerinde bir ileti özelliği ayarlamak, bir ileti özelliği sorgulamak ve bir özelliği ileti tanıtıcısından silmek için bu çağrıları kullanın.

MQCB, MQCB_FUNC, MQCTL
Geri bildirme işlevini kaydettirmek ve denetlemek için bu çağrıları kullanın.

MQSTAT
Önceki zamanuyumsuz koyma işlemlerine ilişkin durum bilgilerini almak için bu çağrıyı kullanın.

MQI çağrılarının bir açıklaması için Çağrı açıklamaları başlıklı konuya bakın.

Eşitleme noktası aramaları

Farklı platformlarda eşitleme noktası çağrıları hakkında bilgi almak için bu bilgileri kullanın.

Eşitleme noktası aramaları aşağıdaki gibi kullanılabilir:

Windows, UNIX ve Linux altyapılarındaIBM WebSphere MQ çağrıları

Linux UNIX Windows

Aşağıdaki ürünler MQCMIT ve MQBACK çağrılarını sağlar:

- PencereleriçinIBM WebSphere MQ
- UNIX and Linux sistemlerindeIBM WebSphere MQ

Son eşitleme noktasından bu yana tüm MQGET ve MQPUT işlemlerinin kalıcı (kesinleştirilmiş) kılınacağı ya da yedekleneceğini kuyruk yöneticisine anlatmak için, programlardaki eşitleme noktası çağrılarını kullanın. CICS ortamında yapılan değişiklikleri kesinleştirmek ve yedeklemek için, EXEC CICS SYNCPOINT ve EXEC CICS SYNCPOINT ROLLBACK gibi komutları kullanın.

Veri dönüştürme, veri tipleri, veri tanımları ve yapılar

İleti Kuyruğu Arabirimi 'ni kullanırken veri dönüştürmeleri, temel veri tipleri, WebSphere MQ veri tanımlamaları ve yapılar hakkında bilgi edinmek için bu bilgileri kullanın.

Veri dönüştürme

MQXCNVC (dönüştürme karakterleri) çağrısı, ileti karakter verilerini bir karakter kümesinden diğerine dönüştürür. Except on WebSphere MQ for z/OS, this call is used only from a data-conversion exit.

MQXCNVC çağrısıyla kullanılan sözdizimine ilişkin MQXCNVC-Karakterlerin dönüştürülmesi başlıklı konuya ve veri dönüştürme çıkışlarının yazılması ve çağrılmasına ilişkin yönergeler için “Veri dönüştürme çıkışları yazılıyor” sayfa 397 başlıklı konuya bakın.

Temel veri tipleri

Desteklenen programlama dilleri için, MQI temel veri tiplerini ya da yapılandırılmamış alanları sağlar.

Bu veri tipleri Temel veri tipleri' ta tam olarak açıklanmıştır.

WebSphere MQ veri tanımlamaları

WebSphere MQ ile sağlanan veri tanımlama dosyaları şunlardır:

- Tüm WebSphere MQ değişmezlerinin ve dönüş kodlarının tanımları
- WebSphere MQ yapılarına ve veri tiplerine ilişkin tanımlar
- Yapıları kullanıma hazırlamak için kullanılan değişmez tanımlamaları
- Çağrıların her biri için işlev prototipleri (yalnızca PL/I ve C dili için)

WebSphere MQ veri tanımı dosyalarının tam açıklaması için bkz. “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77.

Yapılar

“MQI çağrıları” sayfa 188içinde listelenen MQI çağrıları ile kullanılan yapılar, desteklenen programlama dillerinin her biri için veri tanımlama dosyalarında sağlanır.

Yapıların bir özeti için Yapı veri tipleri özeti başlıklı konuya bakın.

IBM WebSphere MQ kod parçası programları ve kitaplık dosyaları

Sağlanan sınırlı kod öbeği programları ve kitaplık dosyaları, her platform için burada listelenir.

Yürütülebilir bir uygulama oluştururken kod parçası programlarının ve kitaplık dosyalarının nasıl kullanılmasıyla ilgili daha fazla bilgi için bkz. “IBM WebSphere MQ uygulaması oluşturulması” sayfa 409. C++ kitaplık dosyalarına bağlanma hakkında bilgi için, bkz. C++ kullanılması *WebSphere MQ Using C++*.

PencereleriçinIBM WebSphere MQ

Pencereleriçin IBM WebSphere MQ işletim sisteminde, işletim sistemi tarafından sağlananlara ek olarak, programınızı, uygulamanızı çalıştırdığınız ortam için sağlanan MQI kitaplık dosyalarına bağlamanız gerekir:

Çizelge 25. Windows uygulamaları için kitaplık dosyaları	
Kitaplık Dosyası	**Ortam**
MQ_INSTALLATION_PATH\Tools\Lib\mqm.lib	Sunucu C (32 bit) için
MQ_INSTALLATION_PATH\Tools\Lib\mqic.lib	C için İstemci (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqmxa.lib	C için sunucu XA arabirimi (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqcxa.lib	C için istemci XA arabirimi (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqicxa.lib	İstemci MTS for C (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqmcics4.lib	C (32 bit) için sunucu TXSeries CICS desteği
MQ_INSTALLATION_PATH\Tools\Lib\mqccics4.lib	C (32 bit) için İstemci TXSeries CICS desteği
MQ_INSTALLATION_PATH\Tools\Lib\mqmzf.lib	C (32 bit) için kurulabilir hizmetler çıkışı
MQ_INSTALLATION_PATH\Tools\Lib\mqmcbb.lib	Server for IBM COBOL (32-bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqmcb.lib	Server for Micro Focus COBOL (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqiccbb.lib	Client for IBM COBOL (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\mqiccb.lib	Client for Micro Focus COBOL (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\imqs23vn.lib	C++ için sunucu (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\imqc23vn.lib	C++ için İstemci (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\imqb23vn.lib	C++ için temel (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib\imqx23vn.lib	C++ için İstemci MTS (32 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqm.lib	Sunucu C (64 bit) için
MQ_INSTALLATION_PATH\Tools\Lib64\mqic.lib	C için İstemci (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqmxa.lib	C için sunucu XA arabirimi (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqcxa.lib	C için istemci XA arabirimi (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqicxa.lib	İstemci MTS for C (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqmcbb.lib	Server for IBM COBOL (64-bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqmcb.lib	Server for Micro Focus COBOL (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqiccbb.lib	Client for IBM COBOL (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\mqiccb.lib	Client for Micro Focus COBOL (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\imqs23vn.lib	C++ için sunucu (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\imqc23vn.lib	C++ için İstemci (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\imqb23vn.lib	C++ için temel (64 bit)
MQ_INSTALLATION_PATH\Tools\Lib64\imqx23vn.lib	C++ için İstemci MTS (64 bit)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

.NET programlarını derlemek için `amqmdnet.dll` kullanın. Ek bilgi için ".NET kullanımı" sayfa 537 kısmındaki " WebSphere MQ .NET programlarının derlenmesi" sayfa 570 konusuna bakın.

Bu dosyalar, önceki yayınlarla uyumluluk için gönderilir:

mqic32.lib
mqic32xa.lib

AIXiçinIBM WebSphere MQ

IBM WebSphere MQ ' ta AIXiçin, programınızı, uygulamanızı çalıştırdığınız ortam için sağlanan MQI kitaplık dosyalarına, işletim sistemi tarafından sağlanan ortamlara bağlamanız gerekir.

İş parçacıklı olmayan bir uygulamada:

Çizelge 26. İş parçacıklı olmayan AIX uygulamaları için kitaplık dosyaları

Kitaplık dosyası	**Ortam**
libmqm.a	C sunucusu için sunucu
libmqic.a & **libmqm.a**	C İçin İstemci
libmqmzf.a	C için kurulabilir hizmet çıkışları
libmqmxa.a	Sunucu XA arabirimi
libmqmxa64.a	Sunucu alternatifi XA arabirimi
libmqcxa.a	İstemci XA arabirimi
libmqcxa64.a	İstemci alternatif XA arabirimi
libmqmcbrt.o	Micro Focus COBOL desteği için WebSphere MQ yürütme ortamı kitaplığı
libmqmcb.a	COBOL için sunucu
libmqicb.a	COBOL için İstemci
libimqc23ia.a	C++ için İstemci
libimqs23ia.a	C++ için sunucu

Bir iş parçacıklı uygulamada:

Çizelge 27. İş parçacıklı AIX uygulamalarına ilişkin kitaplık dosyaları

Kitaplık dosyası	**Ortam**
libmqm_r.a	C sunucusu için sunucu
libmqic_r.a & **libmqm_r.a**	C İçin İstemci
libmqmzf_r.a	C için kurulabilir hizmet çıkışları
libmqmxa_r.a	Sunucu XA arabirimi
libmqmxa64_r.a	Sunucu alternatifi XA arabirimi
libmqcxa_r.a	İstemci XA arabirimi
libmqcxa64_r.a	İstemci alternatif XA arabirimi
libimqc23ia_r.a	C++ için İstemci
libimqs23ia_r.a	C++ için sunucu

HP-UXiçinIBM WebSphere MQ

IBM WebSphere MQ işletim sisteminde, HP-UXiçin, programınızı, uygulamanızı çalıştırdığınız ortam için sağlanan MQI kitaplık dosyalarına, işletim sistemi tarafından sağlanan ortamlara bağlamanız gerekir.

IA64 (IPF) platformu

İş parçacıklı olmayan bir uygulamada:

Çizelge 28. Yivsiz HP-UX uygulamalarına ilişkin kitaplık dosyaları

Kitaplık dosyası	**Ortam**
libmqm.so	C sunucusu için sunucu
libmqic.so & **libmqm.so**	C İçin İstemci
libmqmzf.so	C için kurulabilir hizmet çıkışları
libmqmxa.so	Sunucu XA arabirimi
libmqmxa64.so	Sunucu alternatifi XA arabirimi
libmqcxa.so	İstemci XA arabirimi
libmqcxa64.so	İstemci alternatif XA arabirimi
libimqi23ah.so	C++
libmqmcbrt.o	Micro Focus COBOL desteği için WebSphere MQ yürütme ortamı kitaplığı
libmqmcb.so	COBOL için sunucu
libmqicb.so	COBOL için İstemci

Bir iş parçacıklı uygulamada:

Çizelge 29. İş parçacıklı HP-UX uygulamalarına ilişkin kitaplık dosyaları

Kitaplık dosyası	**Ortam**
libmqm_r.so	C sunucusu için sunucu
libmqmzf_r.so & **libmqm_r.so**	C için kurulabilir hizmet çıkışları
libmqmxa_r.so	Sunucu XA arabirimi
libmqmxa64_r.so	Sunucu alternatifi XA arabirimi
libmqcxa_r.so	İstemci XA arabirimi
libmqcxa64_r.so	İstemci alternatif XA arabirimi
libimqi23ah_r.so	C++

IBM WebSphere MQ - Linux

Linuxiçin IBM WebSphere MQ işletim sisteminde, programınızı, uygulamanızı çalıştırdığınız ortam için sağlanan MQI kitaplık dosyalarına, işletim sistemi tarafından sağlanan ortamlara bağlamanız gerekir.

İş parçacıklı olmayan bir uygulamada:

Çizelge 30. İş parçacıklı olmayan Linux uygulamaları için kitaplık dosyaları

Kitaplık dosyası	**Ortam**
libmqm.so	C sunucusu için sunucu

Çizelge 30. İş parçacıklı olmayan Linux uygulamaları için kitaplık dosyaları (devamı var)	
Kitaplık dosyası	**Ortam**
libmqic.so & **libmqm.so**	C İçin İstemci
libmqmzf.so	C için kurulabilir hizmet çıkışları
libmqmxa.so	Sunucu XA arabirimi
libmqmxa64.so	Sunucu alternatifi XA arabirimi
libmqcxa.so	İstemci XA arabirimi
libmqcxa64.so	İstemci alternatif XA arabirimi
libimqc23gl.so	C++ için İstemci
libimqs23gl.so	C++ için sunucu

Bir iş parçacıklı uygulamada:

Çizelge 31. İş parçacıklı Linux uygulamalarına ilişkin kitaplık dosyaları	
Kitaplık dosyası	**Ortam**
libmqm_r.so	C sunucusu için sunucu
libmqic_r.so & **libmqm_r.so**	C İçin İstemci
libmqmzf_r.so	C için kurulabilir hizmet çıkışları
libmqmxa_r.so	Sunucu XA arabirimi
libmqmxa64_r.so	Sunucu alternatifi XA arabirimi
libmqcxa_r.so	İstemci XA arabirimi
libmqcxa64_r.so	İstemci alternatif XA arabirimi
libimqc23gl_r.so	C++ için İstemci
libimqs23gl_r.so	C++ için sunucu

Solaris içinIBM WebSphere MQ

Solaris 'e ilişkin IBM WebSphere MQ işletim sisteminde, programınızı, uygulamanızı çalıştırdığınız ortam için sağlanan MQI kitaplık dosyalarına, işletim sistemi tarafından sağlanan ortamlara bağlamanız gerekir.

Çizelge 32. Solaris uygulamaları için kitaplık dosyaları	
Kitaplık Dosyası	**Ortam**
libmqm.so	C için sunucu ve istemci
libmqmzse.so	C İçin
libmqic.so	C İçin İstemci
libmqmcs.so	C için ortak hizmetler
libmqmzf.so	C için kurulabilir hizmet çıkışları
libmqmxa.so	Sunucu XA arabirimi
libmqmxa64.so	Sunucu alternatifi XA arabirimi
libmqcxa.so	İstemci XA arabirimi
libmqcxa64.so	İstemci alternatif XA arabirimi

Çizelge 32. Solaris uygulamaları için kitaplık dosyaları (devamı var)	
Kitaplık Dosyası	**Ortam**
libimqc23as.a	C++ için İstemci
libimqs23as.a	C++ için sunucu

Tüm çağrılar için ortak olan parametreler

Tüm çağrılar için ortak iki tip değiştirge vardır: tutamaçlar ve dönüş kodları.

Tutamaçları kullanma

Tüm MQI çağrıları bir ya da daha fazla *çekme noktaları*kullanır. Bu bilgiler, kuyruk yöneticisini, kuyruğu ya da diğer nesneyi, iletiyi ya da aboneliği çağrıya uygun şekilde tanımlar.

Bir programın kuyruk yöneticisiyle iletişim kurması için, programın kuyruk yöneticisini tanıdığı benzersiz bir tanıtıcıya sahip olması gerekir. Bu tanıtıcı, bazen *Hconn*olarak da adlandırılan bir *bağlantı tanıtıcısı*adı olarak adlandırılır. CICS programları için, bağlantı tanıtıcısı her zaman sıfırdır. Diğer tüm altyapılar ya da program biçemleri için, program kuyruk yöneticisine bağlandığında, MQCONN ya da MQCONNX çağrısıyla bağlantı tanıtıcısı döndürülür. Programlar, bağlantı tanıtıcısını, diğer çağrıları kullandıklarında giriş parametresi olarak geçirir.

Bir programın WebSphere MQ nesnesiyle çalışabilmesi için, programın o nesneyi tanıdığı benzersiz bir tanıtıcısı olmalıdır. Bu tanıtıcının adı *nesne tanıtıcısı*, bazen de *Hobj*olarak adlandırılır. Program nesneyi, onunla çalışmak için açtığında, MQOPEN çağrısı bu işleme geri döndürülür. Programlar, nesne tanıtıcısını, sonraki MQPUT, MQGET, MQINQ, MQSET ya da MQCLOSE çağrılarını kullanırken giriş değiştirgesi olarak geçirir.

Benzer şekilde, MQSUB çağrısı, sonraki MQGET, MQCB ya da MQSUBRQ çağrılarında aboneliği tanımlamak için kullanılan bir *subscription handle* ya da *Hsub*döndürür ve bazı çağrı işleme iletisi özellikleri bir *ileti tanıtıcısı* ya da *Hmsg*kullanır.

Dönüş kodlarının anlaşılması

Her çağrıya göre çıkış değiştirgeleri olarak bir tamamlanma kodu ve neden kodu döndürülür. Bunlar toplu olarak *dönüş kodları*olarak bilinir.

Bir çağrıyı başarılı olup olmadığını göstermek için her çağrı, arama tamamlandığında bir *tamamlama kodu* döndürür. Tamamlanma kodu genellikle, başarılı olan MQCC_OK ya da başarısızlığı gösteren MQCC_FAILED olur. Bazı çağrılar bir ara düzey durumu (MQCC_UYARI) döndürebilir, bu da kısmi başarıyı gösterir.

Ayrıca, her çağrı aynı zamanda aramadaki hatanın ya da kısmi başarısının nedenini gösteren bir *neden kodu* döndürür. Bir kuyruğun dolu olması, bir kuyruk için işlemlere izin verilmemesi ve kuyruk yöneticisi için belirli bir kuyruk tanımlanmaması gibi birçok neden kodu vardır. Programlar, işleme nasıl devam edebileceğinize karar vermek için neden kodunu kullanabilir. Örneğin, kullanıcılar kullanıcıların giriş verilerini değiştirmelerini isteyebilir, daha sonra aramayı yeniden yapabilir ya da kullanıcıya bir hata iletisi döndürebilirler.

Tamamlanma kodu MQCC_OK ise, neden kodu her zaman MQRC_NONE olur.

Her çağrıya ilişkin tamamlanma ve neden kodları, bu çağrıya ilişkin açıklamayla listelenir. Çağrı açıklamaları başlıklı konuya bakın ve listeden uygun aramayı seçin.

Düzeltici eylemle ilgili fikirler de içinde olmak üzere daha ayrıntılı bilgi için bkz:

- Diğer tüm WebSphere MQ platformları içinNeden kodları

Arabelleklerin belirtilmesi

Kuyruk yöneticisi, yalnızca gerekli oldukları durumlarda arabellekleri belirtir. Çağrı için bir arabelleğe gerek duymuyorsanız ya da arabelleğin uzunluğu sıfırsa, arabelleğe boş değerli bir gösterge kullanabilirsiniz.

Gereksinim duyduğunuz arabelleğin boyutunu belirlerken her zaman "datallength" değerini kullanın.

Çıkışı bir çağrıdan tutmak için bir arabellek kullandığınızda (örneğin, bir MQGET çağrısına ilişkin ileti verilerini ya da MQINQ çağrısıyla sorgulanan özniteliklerin değerlerini tutmak için), kuyruk yöneticisi belirttiğiniz arabellek geçerli değilse ya da salt okunur bir depoda olduğunda bir neden kodu döndürmeyi dener. Ancak, her zaman bir neden kodu döndüremeyebilir.

UNIX and Linux önemli noktalar

Dikkat etmeniz gereken noktalar.

UNIX and Linux uygulamalarını geliştirirken aşağıdaki noktaları göz önünde bulundurun.

The fork system call in UNIX and Linux systems

Note these considerations when using a fork system call in IBM WebSphere MQ applications.

Uygulamanız `fork`kullanmak isterse, o uygulamanın üst işlemi, IBM WebSphere MQ çağrıları yapmadan önce `fork` çağrısında olmalıdır; örneğin, `MQCONN`, ya da **`ImqQueueManager`**komutunu kullanarak bir IBM WebSphere MQ nesnesi yaratılmalıdır.

Uygulamanız herhangi bir IBM WebSphere MQ çağrısını yaptıktan sonra bir alt işlem yaratmak istiyorsa, uygulama kodu, alt öğenin tam kopyası değil, yeni bir yönetim ortamı olduğundan emin olmak için `exec()` ile bir `fork()` uygulaması kullanmalıdır.

Uygulamanız `exec()`kullanmıyorsa, alt süreç içinde yapılan IBM WebSphere MQ API çağrısı MQRC_ENVIRONMENT_ERROR döndürmesini sağlar.

UNIX and Linux sinyal işleme

Bu, z/OS için WebSphere MQ ya da Pencereleriçin WebSphere MQ için geçerli değildir.

Genel olarak, UNIX, Linux ve IBM i sistemleri iş parçacıklı (süreç) ortamından çok iş parçacıklı bir ortama taşınmış olur. İş parçacıklı ortamda, bazı işlevler yalnızca sinyaller kullanılarak uygulanabilir, ancak çoğu uygulamanın sinyaller ve sinyal işleme konusunda haberdar olması gerekmez. Çok iş parçacıklı ortamda, iş parçacığı tabanlı temel öğeler, sinyaller kullanarak iş parçacıklı ortamlarda uygulanmış olan bazı işlevleri destekler.

Birçok örnekte, sinyaller ve sinyal işleme desteklenir, ancak desteklense de, çok iş parçacıklı ortama ve çeşitli sınırlamalara uygun değildir. Her birinin sinyalleri ele almak için çalıştığı çok iş parçacıklı bir ortamda uygulama kodunu farklı ara katman yazılımı kitaplıklarıyla bütünleştirirken (uygulamanın bir parçası olarak çalışan) bu durum sorunlu olabilir. Bir süreç içinde tek bir yürütme iş parçacığı olduğunda çalışan sinyal işleyicileri (süreç başına tanımlanan) tasarruf ve geri yüklemeye ilişkin geleneksel yaklaşım, çok iş parçacıklı bir ortamda çalışmaz. Bunun nedeni, yürütmenin birçok iş parçacığının, önceden kestirilemeyen sonuçlarla, süreç çapında bir kaynağı saklamaya ve geri yüklemeye çalışabileceği içindir.

İş parçacıklı uygulamalar

Solaris üzerinde, yalnızca tek bir iş parçacığı kullansalar bile, tüm uygulamalar iş parçacıklı olarak değerlendirildiğinden geçerli değildir.

Her MQI işlevi, sinyaller için kendi sinyal işleyicisini ayarlar:

SIGALRM
SIGBUS
SIGFPE
SIGSEGV
SIGILL

Bunlar için kullanıcıların işleyicileri, MQI işlev çağrısının süresine göre değiştirilir. Diğer sinyaller, kullanıcı tarafından yazılan işleyiciler tarafından normal şekilde yakalanabilir. Bir işleyici kurmadıysanız, varsayılan işlemler (örneğin, yoksay, çekirdek dökümü ya da çıkış) yerinde bırakılır.

WebSphere MQ , zamanuyumlu bir sinyal (SIGSEGV, SIGBUS, SIGFPE, SIGILL) işledikten sonra, MQI işlev çağrısını yapmadan önce, kayıtlı herhangi bir sinyal işleyicisine sinyal iletmeyi dener.

İş parçacığı uygulamaları
MQDISC 'ye kadar, bir iş parçacısının MQCONN' dan (ya da MQCONNX) WebSphere MQ ' a bağlı olduğu kabul edilir.

Zamanuyumlu sinyaller

Zamanuyumlu sinyaller belirli bir iş parçacıkında ortaya çıkar.

UNIX and Linux sistemleri, tüm süreç için bu tür sinyaller için bir sinyal işleyicisinin ayarlanmasına güvenli bir şekilde izin verir. Ancak WebSphere MQ , herhangi bir iş parçacığı WebSphere MQ' a bağlı olmakla birlikte uygulama sürecinde aşağıdaki sinyaller için kendi işleyicisini ayarlar:

SIGBUS
SIGFPE
SIGSEGV
SIGILL

Çok iş parçacıklı uygulamalar yazıyorsanız, her sinyal için yalnızca tek bir işlem geniş sinyal işleyicisi vardır. WebSphere MQ , kendi zamanuyumlu sinyal işleyicilerini ayarlarken, her sinyal için önceden kaydedilmiş tüm işleyicileri kaydeder. WebSphere MQ , listelenen sinyallerden birini işledikten sonra, WebSphere MQ , işlem içindeki ilk WebSphere MQ bağlantısı sırasında yürürlükte olan sinyal işleyicisini aramayı dener. Tüm uygulama iş parçacıklarının WebSphere MQile bağlantısı kesildiğinde, önceden kaydedilen işleyiciler geri yüklenir.

Sinyal işleyicileri WebSphere MQtarafından saklandığından ve geri yüklendiğinden, aynı sürecin başka bir iş parçacığının WebSphere MQ' ya da bağlı olması olasılığı varken, uygulama iş parçacıkları bu sinyaller için sinyal işleyicileri oluşturmamalıdır.

Not: Bir uygulama ya da ara katman yazılımı kitaplığı (uygulamanın bir parçası olarak çalışıyorsa), iş parçacığı WebSphere MQ' ya bağlıyken bir sinyal işleyici kurar; uygulamanın sinyal işleyicisinin, bu sinyalin işlenmesi sırasında karşılık gelen WebSphere MQ işleyicisini araması gerekir.

Sinyal işleyicilerini kurarken ve geri yüklerken genel ilke, kaydedilecek son sinyal işleyicinin ilk geri yüklenebilecek ilk kişi olması gerekir:

- Bir uygulama, WebSphere MQ'ya bağlandıktan sonra bir işaret işleyici oluşturduğunda, uygulamanın WebSphere MQ' dan bağlanmadan önce önceki sinyal işleyicisine geri yüklenmesi gerekir.
- When an application establishes a signal handler before connecting to WebSphere MQ, the application must disconnect from WebSphere MQ before restoring its signal handler.

Not: Kaydedilecek son sinyal işleyicisinin, geri yüklenebilecek ilk kişi olması, uygulamada beklenmeyen sinyal işleme ve potansiyel olarak uygulama tarafından sinyallerin kaybedilmesi ile sonuçlanabileceği genel ilkeyi gözlemlememe.

Zamanuyumsuz sinyaller

WebSphere MQ , istemci uygulamaları olmadıkları sürece, iş parçacıklı uygulamalarda zamanuyumsuz sinyaller kullanmaz.

İş parçacıklı istemci uygulamalarına ilişkin ek konular

WebSphere MQ , bir sunucuya G/Ç sırasında aşağıdaki sinyalleri işler. Bu sinyaller, iletişim yığınına göre tanımlanır. Bir iş parçacığı kuyruk yöneticisine bağlıyken, uygulamanın bu sinyaller için bir işaret işleyici oluşturmaması gerekir:

SIGPIPE (TCP/IP için)

Ek konular
UNIX sinyal işleme kullanırken dikkat edilmesi gereken noktalar dikkate alın.

Fastpath (güvenilir) uygulamaları

Fastpath uygulamaları, WebSphere MQ ile aynı işlemde çalışır ve çok iş parçacıklı ortamda çalışır.

Bu ortamda, WebSphere MQ , SIGSEGV, SIGBUS, SIGFPE ve SIGILL zamanuyumlu sinyallerini işler. Diğer tüm sinyaller, WebSphere MQ' a bağlıyken, Fastpath uygulamasına teslim edilmemelidir. Bunun yerine, uygulama tarafından engellenmiş ya da işlenmek zorunda kalmaları gerekir. Bir Fastpath uygulaması böyle bir olayı kesintiye uğratacaksa, kuyruk yöneticisi durdurulmalı ve yeniden başlatılmalı ya da tanımsız bir durumda bırakılabilir. MQCONNX altındaki Fastpath uygulamalarına ilişkin kısıtlamaların tam listesi için bkz. “MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 200.

MQI işlevi, sinyal işleyiciler içinde çağrılar

Siz bir sinyal işleyicisinken, bir MQI işlevi çağırmayın.

Başka bir MQI işlevi etkin durumdayken bir MQI işlevini işaret işleyiciden aramaya çalışırsanız, MQRC_CALL_IN_PROGRESS döndürülür. Başka bir MQI işlevi etkin olmadığında, bir MQI işlevini bir sinyal işleyicisinden çağırmayı denerse, işletim sistemi kısıtlamaları nedeniyle işlem sırasında ya da bir işleyiciden yalnızca seçici çağrıların yayınlanabileceği işletim sistemi kısıtlamalarından dolayı başarısız olur.

Program çıkışı sırasında otomatik olarak çağrılabilecek C++ yok edici yöntemleri için, MQI işlevlerinin çağrılmasına engel olamayabilirsiniz. MQRC_CALL_IN_PROGRESS ile ilgili hataları yoksayın. Bir sinyal işleyici çıkışı çağrılırsa (), WebSphere MQ tutarlılık noktasında kesinleştirilmemiş iletileri her zamanki gibi yedekler ve açık kuyrukları kapatır.

MQI çağrıları sırasında sinyaller

MQI işlevleri kod EINTR ya da uygulama programlarıyla eşdeğer bir kod döndürmez.

Bir MQI çağrısı sırasında bir sinyal oluşursa ve işleyici *return*çağrılırsa, çağrı gerçekleşmemiş gibi çalışmaya devam eder. Özellikle, MQGET işlemi, denetimi hemen uygulamaya döndürmek için bir sinyal ile kesintiye uğratılamaz. Bir MQGET 'den ayrılmak istiyorsanız, kuyruğu GET_DISABLD;' ye ayarlayıp, sonlu bir süre bitimi (MQGMO_WAN gmo.WaitInterval kümesi) ile MQGET çağrısının etrafında bir döngü kullanın ve sinyal işleyicinizi (iş parçacıklı bir ortamda) ya da eşdeğeri bir ortamda, döngüyi bozan bir işaret ayarlamak için eşdeğeri kullanın.

AIX ortamında, WebSphere MQ , sinyaller tarafından kesilen sistem çağrılarının yeniden başlatılmasını gerektirir. İmza eylemiyle (2) kendi sinyal işleyicinizi kurarken, yeni işlem yapısının sa_flags alanındaki SA_RESTART işaretini, tersi durumda WebSphere MQ bir sinyal tarafından kesintiye uğrayan bir çağrıyı tamamlayamayabilir.

Kullanıcı çıkışları ve kurulabilir hizmetler

Çok iş parçacıklı bir ortamda bir WebSphere MQ işleminin bir parçası olarak çalışan kullanıcı çıkışları ve kurulabilir hizmetler, fastpath uygulamalarıyla aynı kısıtlamalara sahiptir. Consider these to be permanently connected to WebSphere MQ and so not using signals or non-threadsafe operating system calls.

VMS çıkış işleyicileri

Kullanıcılar, **SYS$DCLEXH** sistem hizmetini kullanarak bir WebSphere MQ uygulaması için çıkış işleyicileri kurabilir.

Çıkış işleyicisi, bir görüntü çıkışı olduğunda denetimi alır. Normalde Exit ($EXIT) ya da Force Exit ($FORCEX) hizmeti çağırdığınızda görüntü çıkışı gerçekleşir. $FORCEX, hedef işlemi kullanıcı kipinde

kesintiye uğratabiliyor. Daha sonra, tüm kullanıcı kipi çıkış işleyicileri ($DCLEXH ile oluşturulmuş), kuruluş için ters sırada yürütülmeye başlar. Çıkış işleyicileri ve $FORCEX hakkında daha fazla bilgi için *VMS Programming Concepts Manual* ve *VMS System Services Manual*adlı kılavuzlara bakın.

Bir çıkış işleyicisinden bir MQI işlevi çağırırsanız, işlevin işleyişi, görüntünün sona erdirilmesine bağlı olarak değişir. Başka bir MQI işlevi etkinken görüntü sonlandırılırsa, bir `MQRC_CALL_IN_PROGRESS` döndürülür.

Başka bir MQI işlevi etkin değilse ve WebSphere MQ uygulaması için geçersiz kılınmış başka bir MQI işlevi yoksa, bir MQI işlevini bir çıkış işleyicisinden çağırmak olanaklıdır. If upcalls are enabled for the WebSphere MQ application, it fails with the reason code `MQRC_HCONN_ERROR`.

MQCONN ya da MQCONNX çağrısının kapsamı genellikle onu yayınlayan iş parçacığıdır. Çağrılar etkinleştirilirse, çıkış işleyicisi ayrı bir iş parçacığı olarak çalışır ve bağlantı tanıtıcıları paylaşılamaz.

Çıkış işleyicileri, hedef işlemin kesintiye uğratılan bağlamı içinde başlatılır. Bir işleyici tarafından alınan işlemlerin güvenli ve güvenilir olmasını sağlamak, bunların çağrıldığı zamanuyumsuz bir şekilde kesintiye uğratılması için uygulamaya kadar olur.

Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme

WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

Bu bağlantının yapıldığı platform ve programın çalıştığı ortama bağlıdır:

z/OS toplu iş, WebSphere MQ for IBM i, WebSphere MQ on UNIX systems, WebSphere MQ on Linux sistemleri ve WebSphere MQ for Windows

Bu ortamlarda çalışan programlar, bağlanmak için MQCONN MQI çağrısını kullanabilir ve bir kuyruk yöneticisinden bağlantıyı kesmek için MQDISC çağrısına da kullanılabilir. Diğer bir seçenek olarak, programlar MQCONNX çağrısını kullanabilir.

z/OS toplu iş programları, aynı TCB ' de birden çok kuyruk yöneticisine eşzamanlı olarak ya da eşzamanlı olarak bağlanabilir.

IMS

IMS denetim bölgesi başlatıldığında bir ya da daha çok kuyruk yöneticisine bağlanır. Bu bağlantı, IMS komutları tarafından denetlenir. Ancak, ileti kuyruklama IMS programlarının yazıcıları, bağlanmak istedikleri kuyruk yöneticisini belirtmek için MQCONN MQI çağrısını kullanmalıdır. Bu kuyruk yöneticisinden kopmak için MQDISC çağrısını kullanabilirler.

Bir eşitleme noktası oluşturan IMS çağrısının ardından ve başka bir kullanıcı için bir iletiyi işlemeden önce, IMS bağdaştırıcısı, uygulamanın kuyruk yöneticisinden tutamaçları kapatmasını ve bağlantısını kesmesini sağlar.

IMS programları art arda ya da koşutzamanlı olarak aynı TCB ' de kuyruk yöneticilerine bağlanabilir.

CICS Transaction Server for z/OS and CICS for MVS/ESA

CICS programs do not need to do any work to connect to a queue manager because the CICS system itself is connected. Bu bağlantı genellikle başlatma sırasında otomatik olarak yapılır, ancak CKQC işlemini de kullanabilirsiniz.supplied with WebSphere MQ for z/OS.

CICS görevleri, yalnızca CICS bölgesinin kendisinin bağlı olduğu kuyruk yöneticisine bağlanabilir.

Not: CICS programları ayrıca, MQI bağlantısı ve bağlantı kesme çağrılarını (MQCONN ve MQDISC) de kullanabilir. Bunu yapmak isteyebilirsiniz; böylece, bu uygulamaları CICS dışı ortamlar için en az bir recoding (en az) kullanılabilir ortamlarla kaplayabilirsiniz. However, these calls *her zaman* complete successfully in a CICS environment. Diğer bir deyişle, dönüş kodu, kuyruk yöneticiyle bağlantının gerçek durumunu yansıtmayabilir.

Windows ve Open Systems için TXSeries

CICS sisteminin kendisi bağlı olduğu için, bu programların kuyruk yöneticisine bağlanmak için herhangi bir çalışma yapması gerekmez. Bu nedenle, aynı anda yalnızca bir bağlantı desteklenir. CICS

applications must issue an MQCONN call to obtain a connection handle, and an MQDISC call before they exit.

Bir kuyruk yöneticisine bağlanma ve bağlantı kesme hakkında ek bilgi almak için aşağıdaki bağlantıları kullanın:

- “MQCONN çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 199
- “MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 200
- “MQDISC kullanan bir kuyruk yöneticisinden programların bağlantısı kesiliyor” sayfa 205

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

MQCONN çağrısını kullanarak kuyruk yöneticisine bağlanma

MQCONN çağrısını kullanarak bir kuyruk yöneticisine nasıl bağlanılacağını öğrenmek için bu bilgileri kullanın.

Genel olarak, belirli bir kuyruk yöneticisine ya da varsayılan kuyruk yöneticisine bağlanabilirsiniz.

- For IBM WebSphere MQ for z/OS, in the batch environment, the default queue manager is specified in the CSQBDEFV module.
- For IBM WebSphere MQ for Windows, IBM i, UNIX, and Linux systems, the default queue manager is specified in the mqs.ini file.

Diğer bir seçenek olarak, z/OS MVS toplu iş, TSO ve RRS ortamlarında, bir kuyruk paylaşım grubu içindeki herhangi bir kuyruk yöneticisine bağlanabilirsiniz. MQCONN ya da MQCONNX isteği, grubun etkin üyelerinden herhangi birini seçer.

Bir kuyruk yöneticisine bağlandığında, görev için yerel bir kuyruk yöneticisine varmalıdır. Bu, IBM WebSphere MQ uygulaması ile aynı sisteme ait olmalıdır.

In the IMS environment, the queue manager must be connected to the IMS control region and to the dependent region that the program uses. The default queue manager is specified in the CSQQDEFV module when IBM WebSphere MQ for z/OS is installed.

TXSeries CICS ortamı ve Windows ve AIXiçin TXSeries ile, kuyruk yöneticisi CICSolarak bir XA kaynağı olarak tanımlanmalıdır.

Varsayılan kuyruk yöneticisine bağlanmak için, tümüyle boşluklardan oluşan ya da boş değerli (X'00 ') karakterlerle başlayan bir ad belirterek MQCONN' yi çağırın.

Bir uygulamanın kuyruk yöneticisine başarıyla bağlanması için bir uygulama yetkilendirilmelidir. Ek bilgi için Security(Güvenlik) başlıklı konuya bakın.

MQCONN ' den çıkış:

- Bir bağlantı tanıtıcısı (**Hconn**)
- Tamamlanma kodu
- Neden kodu

İzleyen MQI çağrılarında bağlantı tanıtıcısını kullanın.

Neden kodu, uygulamanın o kuyruk yöneticisine önceden bağlı olduğunu gösteriyorsa, döndürülen bağlantı tanıtıcısı, uygulama ilk kez bağlandığında döndürülen bağlantı tanıtıcısı ile aynıdır. Çağıran uygulamanın bağlı kalmasını beklediğinden, uygulama bu durumda MQDISC çağrısını yayınlamamalıdır.

Bağlantı tanıtıcısı kapsamı, nesne tutamacının kapsamı ile aynıdır (bkz. “MQOPEN çağrısını kullanarak nesnelerin açılması” sayfa 207).

Parametrelere ilişkin açıklamalar, MQCONNiçindeki MQCONN çağrısının tanımında verilmiştir.

Kuyruk yöneticisi çağrıyı yayınlarken ya da kuyruk yöneticisi kapatılıyorsa, MQCONN çağrısı başarısız olur ya da kuyruk yöneticisi durdurulmuş durumdaysa başarısız olur.

MQCONN ya da MQCONNX kapsamı

MQCONN ya da MQCONNX çağrısının kapsamı genellikle onu yayınlayan iş parçacığıdır. Yani, çağrıdan döndürülen bağlantı tanıtıcısı yalnızca çağrıyı yayınlayan iş parçacığın içinde geçerlidir. Tutamacı kullanarak herhangi bir zamanda yalnızca bir arama yapılabilir. Farklı bir iş parçacığıdan kullanılırsa, geçersiz olarak reddedilir. Uygulamanınızda birden çok iş parçacığınız varsa ve her biri IBM WebSphere MQ çağrısını kullanmak isterse, her biri MQCONN ya da MQCONNX yayınlamalıdır.

Bir işlem birden çok MQCONN çağrısını yaptığında, aynı kuyruk yöneticisine her çağrı için bu gerekli değildir. However, only one WebSphere MQ connection can be made from a thread at a time. Diğer bir seçenek olarak, “Shared (thread independent) connections with MQCONNX” sayfa 203 ' un tek bir iş parçacıktan birden çok WebSphere MQ bağlantısına ve herhangi bir iş parçacığıdan WebSphere MQ bağlantısının kullanılmasına izin vermesi için de göz önünde bulundurun.[1]

Uygulamanız istemci olarak çalışıyorsa, iş parçacığı içinde birden çok kuyruk yöneticisine bağlanabiliyor.

MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma

MQCONNX çağrısı MQCONN çağrısına benzer, ancak arama çalışmalarının yolunu denetleme seçeneklerini içerir.

MQCONNX 'e giriş olarak, bir kuyruk yöneticisi adı ya da z/OS paylaşılan kuyruk sistemlerinde kuyruk paylaşım grubu adı girebilirsiniz. MQCONNX 'in çıkışı şöyledir:

- Bir bağlantı tanıtıcısı (Hconn)
- Tamamlanma kodu
- Neden kodu

İzleyen MQI çağrılarında bağlantı tanıtıcısını kullanıyorsunuz.

MQCONNX 'in tüm değiştirgelerinin tanımı MQCONNXiçinde verilmiştir. *Options* alanı, MQCNO ' nun herhangi bir sürümü için STANDARD_BINDING, FASTPATH_BINDING, SHARED_BINDING ya da ISOLATED_BINDING olarak ayarlamanızı sağlar. Ayrıca, bir MQCONNX çağrısı kullanarak paylaşılan (iş parçacığı bağımsız) bağlantıları da yapabilirsiniz. Bunlarla ilgili daha fazla bilgi için bkz. “Shared (thread independent) connections with MQCONNX” sayfa 203 .

[1] When using multithreaded applications with IBM WebSphere MQ on UNIX and Linux systems you need to ensure that the applications have a sufficient stack size for the threads. Çok iş parçacıklı uygulamalar kendi başına ya da diğer sinyal işleyicilerle (örneğin, CICSgibi) MQI çağrıları yaparken 256 KB ya da daha büyük bir yığın boyutu kullanmayı düşünün.

MQCNO_STANDARD_BINDING

Varsayılan olarak, MQCONNX (MQCONN gibi), WebSphere MQ uygulamasının ve yerel kuyruk yöneticisi aracısının ayrı süreçlerde çalıştırıldığı iki mantıksal iş parçacığını belirtir. The WebSphere MQ application requests the WebSphere MQ operation and the local queue manager agent services the request. Bu, MQCONNX çağrısında MQCNO_STANDARD_BINBING seçeneği tarafından tanımlanır.

If you specify MQCNO_STANDARD_BINDING, the MQCONNX call uses either MQCNO_SHARED_BINDING or MQCNO_ISOLATED_BINDING, depending on the value of the DefaultBindType attribute of the queue manager, which is defined in qm.ini or the Pencereler registry.

Bu varsayılan değerdir.

`mqm` kitaplığına bağlantı oluşturuyorsanız, önce varsayılan bağ tanımlama tipini kullanan bir standart sunucu bağlantısı girişiminde bulunulması gerekir. Temeldeki sunucu kitaplığı yüklenemediyse, bunun yerine bir istemci bağlantısı girişiminde bulunulması gerekir.

- MQ_CONNECT_TYPE ortam değişkeni belirtilirse, MQCB_STANDARD_BINDING belirtilirse, MQCONN ya da MQCONNX 'in davranışını değiştirmek için aşağıdaki seçeneklerden biri sağlanabilir. (Bu kural dışı durum, MQCNO_FASTPATH_BINDING MQ_CONNECT_TYPE ile LOCAL ya da STANDARD değerine ayarlandıysa, uygulama için ilgili bir değişiklik yapılmaksızın, fastpath bağlantılarının yönetici tarafından düşürülmesine izin verir:

Değer	**Anlamı**
CLIENT	Yalnızca istemci bağlantısı deneniyor.
FastPath	Bu değer önceki yayınlarda desteklendi, ancak belirtilirse yoksayılır.
LOCAL	Yalnızca sunucu bağlantısı denendi. Fastpath bağlantıları, standart bir sunucu bağlantısına düşürülebiliyor.
Standart	Önceki yayın düzeyleriyle uyumluluk için desteklenir. Bu değer şimdi LOCAL olarak ele alınır.

- MQCONN çağrıldığında MQ_CONNECT_TYPE ortam değişkeni ayarlanmazsa, varsayılan bağ tanımlama tipini kullanan bir standart sunucu bağlantısı girişiminde bulunulması denenir. Sunucu kitaplığı yüklenemezse, bir istemci bağlantısı girişiminde bulunmaya çalışılır.

MQCNO_FASTPATH_BINDING

Güvenilen uygulamalar , WebSphere MQ uygulamasının ve yerel kuyruk yöneticisi aracısının aynı işlem haline geldiğini belirtir. Aracı sürecinin kuyruk yöneticisine erişmek için artık bir arabirimi kullanması gerekmediği için, bu uygulamalar kuyruk yöneticisinin bir uzantısı haline gelir. Bu, MQCONNX çağrısında MQCNO_FASTPATH_BINDING seçeneği tarafından tanımlanır.

Güvenilen uygulamaları iş parçacıklı WebSphere MQ kitaplıklarına bağlamaya gerek duyarsınız. Bir WebSphere MQ uygulamasının güvenilir olarak çalıştırılmasına ilişkin yönergeler için bkz. MQCNO Seçenekleri.

Bu seçenek en yüksek başarımı sağlar.

Not: Bu seçenek, kuyruk yöneticisinin bütünlüğünü sağlar: depolamanın üzerine yazıldığında koruma yoktur. Bu, uygulama iletileri ve kuyruk yöneticisinde bulunan diğer verileri de gösterebilecek hatalar içeriyorsa geçerlidir. Bu seçeneği kullanmadan önce bu sorunları göz önünde bulundurun.

MQCNO_SHARED_BINDING

Uygulamayı ve yerel kuyruk yöneticisi aracısını ayrı süreçlerde çalıştırabilmek için bu seçeneği belirleyin. Bu, kuyruk yöneticisinin bütünlüğünü korur; yani, kuyruk yöneticisini errant programlarından korur. Ancak, uygulama ve yerel kuyruk yöneticisi aracıları bazı kaynakları paylaşır.

Bu seçenek, MQCNO_FASTPATH_BINDING ve MQCNO_ISOLATED_BINDING arasında, hem kuyruk yöneticisinin bütünlüğünü korumak açısından, hem de MQI çağrılarının başarımı açısından ara ara sağlar.

Kuyruk yöneticisi bu bağ tanımı tipini desteklemiyorsa, MQCNO_SHARED_BINDING yoksayılır. Bu seçenek belirlenmemiş gibi işleme devam eder.

Bir uygulama, MQCNO_SHARED_BINDING MQCNO_SHARED_BINDING komutunu kullanarak yerel kuyruk yöneticisine bağlıysa, uygulama çalışırken kuyruk yöneticisi durdurulabilir. Uygulama çalışmaya devam ederken kuyruk yöneticisini yeniden başlatırken, kuyruk yöneticisi tarafından gerekli olan kaynaklar üzerinde çalışmaya devam ettikçe, kuyruk yöneticisini başlatma girişimi AMQ7018 hatasıyla başarısız olur.

Kuyruk yöneticisini başlatmak için uygulamayı durdurmanız gerekir.

MQCNO_ISOLATED_BINDING

Uygulamayı ve yerel kuyruk yöneticisi aracısını MQCNO_SHARED_BINDING MQCNO_SHARED_BINDING için ayrı işlemlerde çalıştırırken yapmak için bu seçeneği belirleyin. Ancak bu durumda, uygulama işlemi ve yerel kuyruk yöneticisi aracısı, kaynakları paylaşmadıkları için birbirlerinden yalıtılır.

Bu, kuyruk yöneticisinin bütünlüğünü korumak için en güvenli seçenektir, ancak bu, MQI çağrılarının en yavaş başarımını sağlar.

Kuyruk yöneticisi bu bağ tanımlama tipini desteklemiyorsa, MQCNO_ISOLATED_BINDING yoksayılır. Bu seçenek belirlenmemiş gibi işleme devam eder.

MQCNO_CLIENT_BINDING

Uygulamanın yalnızca istemci bağlantısı denemesini sağlamak için bu seçeneği belirleyin. Bu seçenek aşağıdaki sınırlamalara sahiptir:

- MQCNO_CLIENT_BINDING, MQRC_OPTIONS_ERROR ile z/OS üzerinde reddedildi.
- MQCNO_CLIENT_BINDING, MQCNO_STANDARD_BINDING dışında herhangi bir MQCNO bağ tanımı seçeneğiyle belirtildiyse, MQRC_OPTIONS_ERROR ile reddedildi.
- MQCNO_CLIENT_BINDING, bağ tanımlama tipini seçmek için kendi yöntemlerine sahip olduğu için Java için kullanılamaz.
- V 7.5.0.7 IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 7öncesinde, MQCNO_CLIENT_BINDING, bağ tanımlama tipini seçmek için kendi mekanizmalarına sahip olduğu için .NET için kullanılabilir değildir. Version 7.5.0, Fix Pack 7' tan, MQCNO_CLIENT_BINDING için .NET kullanmaya ilişkin kısıtlama kaldırıldı.
- MQCONNX çağrıldığında MQ_CONNECT_TYPE ortam değişkeni ayarlanmazsa, varsayılan bağ tanımlama tipini kullanan bir standart sunucu bağlantısı girişiminde bulunulması denenir. Sunucu kitaplığı yüklenemezse, bir istemci bağlantısı girişiminde bulunmaya çalışılır.

MQCNO_LOCAL_BINBINA

Uygulamanın bir sunucu bağlantısını denemesini sağlamak için bu seçeneği belirleyin. MQCNO_FASTPATH_BINDING, MQCNO_ISOLATED_BINDING ya da MQCNO_SHARED_BINBING değeri de belirtilirse, bağlantı bu tipte olur ve bu bölümde belgelenir. Tersi durumda, varsayılan bağ tanımlama tipi kullanılarak standart bir sunucu bağlantısı girişiminde bulunmaya çalışılır. MQCNO_LOCAL_BINBING, aşağıdaki sınırlamalara sahiptir:

- MQCNO_LOCAL_BINDING, z/OSüzerinde yoksayıldı.
- MQCNO_RECONNECT_AS_DEF dışında herhangi bir MQCNO yeniden bağlanma seçeneğiyle belirtilirse, MQCNO_LOCAL_BINBING, MQRC_OPTIONS_ERROR ile reddedilir.
- MQCNO_LOCAL_BINDING, bağ tanımlama tipini seçmek için kendi mekanizmalarına sahip olduğu için, Java için kullanılamaz.
- V 7.5.0.7 IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 7öncesinde, MQCNO_LOCAL_BINDING, bağ tanımlama tipini seçmek için kendi mekanizmalarına sahip olduğu

için .NET için kullanılamaz. Version 7.5.0, Fix Pack 7' tan, MQCNO_LOCAL_BINDING için .NET kullanmaya ilişkin kısıtlama kaldırıldı.

- MQCONNX çağrıldığında MQ_CONNECT_TYPE ortam değişkeni ayarlanmazsa, varsayılan bağ tanımlama tipini kullanan bir standart sunucu bağlantısı girişiminde bulunulması denenir. Sunucu kitaplığı yüklenemezse, bir istemci bağlantısı girişiminde bulunmaya çalışılır.

z/OS üzerinde bu seçenekler tolere edilir, ancak yalnızca standart bağlı bir bağlantı gerçekleştirilir. MQCNO Sürüm 3, z/OSiçin dört alternatif seçeneğe izin verir:

MQCNO_SERIALIZE_CONN_TAG_QSG

Bu, bir uygulamanın, bir uygulamanın yalnızca bir eşgörünümünün, bir kuyruk paylaşım grubunda herhangi bir zamanda çalıştırılmasını istemesine olanak sağlar. Bu, uygulama tarafından belirtilen ya da türetilen bir değere sahip bir bağlantı etiketinin kullanımını kayda geçirilerek elde edilir. Etiket, Sürüm 3 MQCNO ' da belirtilen 128 baytlık karakter dizilimidir.

MQCNO_RESTRICT_CONN_TAG_QSG

Bu, her biri bir kuyruk yöneticisine bağlanabilen birden çok süreçten (ya da bir TCB) oluşan bir uygulamanın bulunduğu bir yerde kullanılır. Yalnızca, etiketin geçerli bir kullanımı yoksa ya da istekte bulunan uygulama aynı işlem kapsamı içindeyse bağlantıya izin verilir. Bu, etiket sahibiyle aynı kuyruk paylaşım grubunda bulunan MVS adres alanıdır.

MQCNO_SERIALIZE_CONN_TAG_Q_MGR

Bu, MQCNO_SERIALIZE_CONN_TAG_QSG komutuna benzer, ancak istenen etiketin zaten kullanımda olup olmadığını görmek için yalnızca yerel kuyruk yöneticisi sorgulanır.

MQCNO_RESTRICT_CONN_TAG_Q_MGR

Bu, MQCNO_RESTRICT_CONN_TAG_QSG komutuna benzer; ancak, istenen etiketin kullanımda olup olmadığını görmek için yalnızca yerel kuyruk yöneticisi sorgulanır.

Güvenilir uygulamalara ilişkin kısıtlamalar

Güvenilen uygulamalar için aşağıdaki kısıtlamalar geçerlidir:

- Güvenilir uygulamaların, kuyruk yöneticisinden belirtik olarak bağlantısını kesmeniz gerekir.
- Kuyruk yöneticisini endmqm komutuyla sona erdirmeden önce güvenilir uygulamaları durdurmanız gerekir.
- Zamanuyumsuz sinyalleri ve zamanlayıcı kesintilerini (`sigkill`gibi) MQCNO_FASTPATH_BINDING ile birlikte kullanmamalısınız.
- Tüm altyapılarda, aynı işlemdeki başka bir iş parçacığı farklı bir kuyruk yöneticisine bağlıyken, güvenilir bir uygulama içindeki bir iş parçacığı kuyruk yöneticisine bağlanamaz.
- On WebSphere MQ on UNIX and Linux systems you must use mqm as the effective userID and groupID for all MQI calls. MQI olmayan bir çağrı yapmadan önce bu tanıtıcıları değiştirebilirsiniz (örneğin, bir dosya açmak gibi), ancak sonraki MQI çağrısını yapmadan önce, bu dosyayı mqm 'ye geri çevirmeniz *gerekir* .
- WebSphere MQ for HP-UXüzerinde, çok iş parçacıklı hızlı yol uygulamalarının varsayılan değerden daha büyük bir yığın boyutu ayarlamaya gerek vardır. 256 KB ' lik bir boyut kullanın.
- On WebSphere MQ for Windows trusted 64-bit applications are not supported. Güvenilen bir 64 bit uygulamayı çalıştırmayı denerse, bu uygulama standart bir bağlantı bağlantısına indirgenir.
- On WebSphere MQ on UNIX and Linux systems trusted 32-bit applications are not supported. Güvenilen bir 32 bit uygulamayı çalıştırmayı denerse, bu uygulama standart bir bağ bağlantısı düzeyine indirgenir.

Shared (thread independent) connections with MQCONNX

MQCONNX ile paylaşılan bağlantılar ve dikkate alınacak bazı kullanım notları hakkında bilgi edinmek için bu bilgileri kullanın.

Not: Not supported on WebSphere MQ for z/OS.

WebSphere MQ for z/OSdışındaki WebSphere MQ platformlarında, MQCONN ile yapılan bir bağlantı yalnızca bağlantıyı yapan iş parçacığıyla kullanılabilir. MQCONNX çağrısına ilişkin seçenekler, bir işlemdeki tüm iş parçacıkları tarafından paylaşılabilen bir bağlantı yaratmanızı sağlar. Uygulamanız, aynı iş parçacığıda MQI çağrısını gerektiren bir işlemsel ortamda çalışıyorsa, aşağıdaki varsayılan seçeneği kullanmanız gerekir:

MQCNO_HANDLE_SHARE_NONE
Paylaşılmayan bir bağlantı oluşturur.

Diğer birçok ortamda, aşağıdaki iş parçacığı bağımsız, paylaşılan bağlantı seçeneklerinden birini kullanabilirsiniz:

MQCNO_HANDLE_SHARE_BLOCK
Paylaşılan bir bağlantı oluşturur. Bir MQCNO_HANDLE_SHARE_BLOCK bağlantısında, bağlantı şu anda başka bir iş parçacığıdaki bir MQI çağrısı tarafından kullanıldıysa, MQI çağrısı, yürürlükteki MQI çağrısı tamamlanıncaya kadar bekler.

MQCNO_HANDLE_SHARE_NO_BLOCK
Paylaşılan bir bağlantı oluşturur. On a MQCNO_HANDLE_SHARE_NO_BLOCK connection, if the connection is currently in use by an MQI call on another thread, the MQI call fails immediately with a reason of MQRC_CALL_IN_PROGRESS.

MTS (Microsoft Transaction Server) ortamı dışında, varsayılan değer MQCNO_HANDLE_SHARE_NONEdeğeridir. MTS ortamında varsayılan değer MQCNO_HANDLE_SHARE_BLOCKdeğeridir.

MQCONNX çağrısından bir bağlantı tanıtıcısı döndürülür. Tanıtıcı, süreçteki herhangi bir iş parçacığıdaki sonraki MQI çağrıları tarafından kullanılabilir ve bu çağrıları MQCONNX' den döndürülen tanıtıcı ile ilişkilendirir. Tek bir paylaşılan tanıtıcı kullanılarak yapılan MQI çağrıları iş parçacıkları arasında diziselleştirilir.

Örneğin, paylaşılan bir tanıtıcı ile aşağıdaki etkinlik sırası mümkündür:

1. İş parçacığı 1 sorunları MQCONNX ve paylaşılan bir tanıtıcı alır *h1*
2. İş parçacığı 1, bir kuyruğu açar ve *h1*kullanarak bir alma isteği yayınlar
3. İş parçacığı 2, *h1*kullanarak bir put isteği yayınlar.
4. İş parçacığı 3, *h1*kullanarak bir put isteği yayınlar.
5. İş parçacığı 2, *h1*komutunu kullanarak MQDISC

Tanıtıcı herhangi bir iş parçacığının kullanımında olmakla birlikte, bağlantıya erişim diğer iş parçacıklarına kullanılamaz. In circumstances where it is acceptable that a thread waits for any previous call from another thread to complete, use MQCONNX with the option MQCNO_HANDLE_SHARE_BLOCK.

Ancak engelleme, zorluklara neden olabilir. “2” sayfa 204adımında, 1 numaralı iş parçacığıda henüz gelmemiş olabilecek iletiler için bekleyen bir alma isteği yayınlar (bekleme ile alma). Bu durumda, 2. ve 3 numaralı iş parçacıkları, iş parçacığının 1 numaralı iş parçacığının alma isteği kadar uzun süre bekletilmeye (engellendi) bırakılır. Tanıtıcıda başka bir MQI çağrısı zaten çalışıyorsa, bir MQI çağrısının bir hata ile döndürülemesini tercih ederseniz, MQCONNX seçeneğini MQCNO_HANDLE_SHARE_NO_BLOCKseçeneğiyle kullanın.

Paylaşılan bağlantı kullanım notları

1. Bir nesne açılarak yaratılan herhangi bir nesne tanıtıcısı (Hobj) bir Hconn ile ilişkilendirilir; bu nedenle, paylaşılan bir Hconn için, Hobjs, Hconn kullanan herhangi bir iş parçacığı tarafından da paylaşılır ve kullanılabilir. Benzer şekilde, bir Hconn altında başlatılan herhangi bir iş birimi Hconn ile ilişkilendirilir; dolayısıyla, bu işlem, paylaşılan Hconn ile iş parçacıkları arasında paylaşılır.
2. *Herhangi biri* iş parçacığı, paylaşılan bir Hconn 'un bağlantısını kesmek için MQDISC ' yi çağırabilir; yalnızca ilgili MQCONNX 'i arayan iş parçacıkından değil. MQDISC, tüm iş parçacıklarının kullanılamaması için Hconn 'ı sona erdirir.
3. Tek bir iş parçacığı birden çok paylaşılan Hconn özelliğini kullanabilir; örneğin, bir iletiyi paylaşılan bir Hconn altına koymak için MQPUT kullanarak, başka bir paylaşılan Hconn kullanarak başka bir ileti daha koyun; her işlem farklı bir yerel iş birimi altında olur.

4. Paylaşılan Hconns, genel bir iş birimi içinde kullanılamaz.

MQ_CONNECT_TYPE ile MQCONNX çağrı seçeneklerinin kullanılması

MQ_CONNECT_TYPE ile nasıl kullanıldıklarını anlamak için bu bilgileri kullanın.

WebSphere MQ for IBM i, WebSphere MQ for Windowsve WebSphere MQ on UNIX and Linux sistemlerinde, MQCONNX çağrısında kullanılan MQCNO yapısının *Options* alanında belirtilen bağ tanımı tipiyle birlikte MQ_CONNECT_TYPE ortam değişkenini kullanabilirsiniz.

Çizelge 33. MQ_CONNECT_TYPE ortam değişkeni

MQCONNX çağrı seçeneği	**MQ_CONNECT_TYPE ortam değişkeni**	**Sonuç**
Standart	Tanımlı değil	Standart
Standart	Standart	Standart
Standart	FastPath	Standart
Standart	CLIENT	CLIENT
Standart	LOCAL	Standart

MQCNO_STANDARD_BINDING belirtilmediyse, varsayılan olarak MQCNO_STANDARD_BINDING değerini kullanan MQCNO_NONE değerini kullanabilirsiniz.

MQDISC kullanan bir kuyruk yöneticisinden programların bağlantısı kesiliyor

MQDISC kullanarak bir kuyruk yöneticisinden programların bağlantısını kesme hakkında bilgi edinmek için bu bilgileri kullanın.

MQCONN ya da MQCONNX çağrısını kullanan bir kuyruk yöneticisine bağlı bir program, kuyruk yöneticisiyle tüm etkileşimi bitirdiğinde, MQDISC çağrısını kullanarak bağlantıyı keser.

- CICS Transaction Server for z/OS uygulamaları üzerinde, MQCONNX kullanılmadığı ve uygulama sona erdirilmeden önce bağlantı etiketini atmak istiyorsanız, bu çağrıların isteğe bağlı olduğuwhereuygulamaları.
- On WebSphere MQ for IBM i where, when you sign off from the operating system, an implicit MQDISC call is made.

MQDISC çağrısına giriş olarak, kuyruk yöneticisine bağlandığınızda MQCONN ya da MQCONNX tarafından döndürülen bağlantı tanıtıcısını (Hconn) sağlamanız gerekir.

Except on CICS on z/OS, after MQDISC is called the connection handle (Hconn) is no longer valid, and you cannot issue any further MQI calls until you call MQCONN or MQCONNX again. MQDISC, bu tanıtıcıyı kullanarak hala açık olan nesneler için örtük bir MQCLOSE yapar.

z/OSiçin WebSphere MQ ' e bağlanmak üzere MQCONNX 'i kullanırsanız, MQDISC, MQCONNX tarafından kurulan bağlantı etiketinin kapsamını da sona erdirir. Ancak, bir CICS, IMSya da RRS uygulamasında, bir bağlantı etiketiyle ilişkilendirilmiş etkin bir kurtarma birimi varsa, MQDISC, MQRC_CONN_TAG_NOT_SAILD neden koduyla reddedilir.

Değiştirgelere ilişkin açıklamalar, MQDISCiçindeki MQDISC çağrısına ilişkin açıklamayla verilir.

MQDISC komutu verilmediği zaman

Bir standart, paylaşılmayan bağlantı (Hconn), oluşturma iş parçacığı sona erdiğinde temizlenir. Paylaşılan bir bağlantı, tüm işlem sona erdiğinde örtük olarak geriletilir ve bağlantısı kesilir. Hconn hala varsa, paylaşılan Hconn 'ı yaratan iş parçacığı sonlandırılırsa, Hconn hala kullanılabilir durumda olur.

Yetki denetimi

MQCLOSE ve MQDISC çağrıları genellikle hiçbir yetki denetimi gerçekleştirmez.

Bir WebSphere MQ nesnesi açma ya da bağlanma yetkisi olan bir işin olağan bir şekilde, o nesneden kapanması ya da bu nesneden bağlantısını kesmeniz gerekir. Bir WebSphere MQ nesnesini açmış ya da açmış bir işin yetkisi iptal edilse bile, MQCLOSOSE ve MQDISC çağrıları kabul edilir.

Nesnelerin açılması ve kapatılması

Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

Aşağıdaki işlemlerden herhangi birini gerçekleştirmek için, ilgili WebSphere MQ nesnesini *açmalısınız* .

- İletileri kuyruğa koy
- Kuyruktan ileti alma (gözatma ya da alma)
- Bir nesnenin özniteliklerini ayarlama
- Herhangi bir nesnenin özniteliklerine ilişkin olarak sorgulama

Nesneyi açmak için, çağrıya ilişkin seçenekleri kullanarak nesneyle ne yapmak istediğinizi belirtmek için MQOPEN çağrısını kullanın. Tek kural dışı durum, kuyruğa tek bir ileti koymak istiyorsanız, kuyruğu hemen kapatmanız gerekir. Bu durumda, MQPUT1 çağrısını kullanarak *açma* aşasını atlayabilirsiniz (bkz. “MQPUT1 çağrısını kullanarak bir ileti kuyruğa konması” sayfa 224).

MQopen çağrısını kullanarak bir nesneyi açmadan önce, programınızı bir kuyruk yöneticisine bağlamanız gerekir. Bu, “Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198' ta tüm ortamlar için ayrıntılı olarak açıklanmıştır.

Açabileceğiniz dört tip WebSphere MQ nesnesi vardır:

- Kuyruk
- Ad Listesi
- Süreç tanımlaması
- Kuyruk yöneticisi

Tüm bu nesneleri, MQOPEN çağrısını kullanarak benzer şekilde açmanızı sağlar. WebSphere MQ nesnelerine ilişkin ek bilgi için Nesnelerkonusuna bakın.

Aynı nesneyi bir kereden fazla açabilirsiniz ve her defasında yeni bir nesne tanıtıcısı elde edebilirsiniz.Bir kuyruktaki iletilere bir tanıtıcı kullanarak göz atmak ve aynı kuyruktan iletileri başka bir tanıtıcı kullanarak kaldırmak isteyebilirsiniz. Bu işlem, aynı nesneyi kapatmak ve yeniden açmak için kaynakları kullanarak kurtarır. Ayrıca, aynı anda iletileri kaldırmak için *ve* ' e göz atmak için bir kuyruk da açabilirsiniz.

Ayrıca, tek bir MQXX_ENCODE_CASE_ONE open ile birden çok nesne açabilir ve bu nesneleri MQCLOSE kullanarak kapatabilirsiniz. Bunun nasıl yapacağına ilişkin bilgi için bkz. “Dağıtım listeleri” sayfa 225 .

Bir nesneyi açmaya çalıştığınızda, kuyruk yöneticisi, o nesneyi MQOPEN çağrısında belirlediğiniz seçenekler için açma yetkisine sahip olduğunuzu denetler.

Bir program kuyruk yöneticisinden bağlantıyı kestiğinde nesneler otomatik olarak kapatılır. IMS ortamında, bir GU (benzersiz alma) IMS çağrısı sonrasında yeni bir kullanıcı için bir program işlenmeye başladığında bağlantı kesilmeye zorlanır. IBM i platformunda, bir iş sona erdiğinde nesneler otomatik olarak kapatılır.

Açtığınız nesneleri kapatmak için iyi bir programlama uygulamasıdır. Bunu yapmak için MQCLOSE çağrısını kullanın.

Nesneleri açma ve kapatma hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “MQOPEN çağrısını kullanarak nesnelerin açılması” sayfa 207
- “Dinamik kuyruklar oluşturma” sayfa 214
- “Uzak Kuyrukların Açılması” sayfa 214
- “MQCLOSE çağrısını kullanarak nesneler kapatılıyor” sayfa 215

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187

Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

"Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme" sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

"İletileri Kuyruğa Koyma" sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

"Kuyruktan Ileti Alınması" sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

"Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme" sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

"İş birimlerinin kesinleştirilmesi ve yedeklenmesi" sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

"Starting IBM WebSphere MQ applications using triggers" sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

"MQI ve kümelerle çalışma" sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

MQOPEN çağrısını kullanarak nesnelerin açılması

MQOPEN çağrısını kullanarak nesneleri açma hakkında bilgi edinmek için bu bilgileri kullanın.

MQOPER çağrısına giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı. z/OSüzerindeki CICS uygulamaları için, değişmez MQHC_DEF_HCONN (sıfır değerine sahip) değerini belirtebilir ya da MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanabilirsiniz. Diğer programlar için, her zaman MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanın.
- Nesne tanımlayıcı yapısını (MQOD) kullanarak açmak istediğiniz nesneye ilişkin bir açıklama.
- Çağrıyı denetleyen bir ya da daha fazla seçenek.

MQXX_ENCODE_CASE_ONE open komutunun çıkışı şöyledir:

- Nesneye erişiminizi gösteren bir nesne tanıtıcısı. Bunu izleyen herhangi bir MQI çağrısına giriş sırasında kullanın.
- Dinamik bir kuyruk yaratıyorsanız, değiştirilmiş bir nesne tanımlayıcı yapısı (ve platformunuzda destekleniyorsa).
- Bir tamamlanma kodu.
- Bir neden kodu.

Bir nesne tutamacının kapsamı

Bir nesne tanıtıcısı (Hobj) kapsamı, bağlantı tanıtıcısı (Hconn) kapsamı ile aynıdır.

Bu, "MQCONN ya da MQCONNX kapsamı" sayfa 200 ve "Shared (thread independent) connections with MQCONNX" sayfa 203ile kaplıdır. Ancak, bazı ortamlarda dikkate alınması gereken noktalar vardır:

CICS
Bir CICS programında, tanıtıcıyı yalnızca, MQOPEN çağrısını yaptığınız aynı CICS görevi içinde kullanabilirsiniz.

IMS ve z/OS toplu işi
IMS ve toplu iş ortamlarında, aynı görev içindeki tanıtıcıyı kullanabilir, ancak herhangi bir alt görev içinde kullanamazsınız.

Descriptions of the parameters of the MQOPEN call are given in MQOPEN.

Aşağıdaki kısımlarda, MQOPEN ' a giriş olarak sağlamanız gereken bilgiler açıklanmaktadır.

Nesnelerin tanımlanması (MQOD yapısı)

Açmak istediğiniz nesneyi tanımlamak için MQOD yapısını kullanın. Bu yapı, MQOPEN çağrısına ilişkin bir giriş parametresidir. (Devingen kuyruk yaratmak için MQOPEN çağrısını kullandığınızda, yapı kuyruk yöneticiyle değiştirilir.)

MQOD yapısıyla ilgili tüm ayrıntılar için MQODbaşlıklı konuya bakın.

Dağıtım listeleri için MQOD yapısının kullanılmasıyla ilgili bilgi için, “Dağıtım listeleri” sayfa 225altındaki “MQOD yapısının kullanılması” sayfa 226 başlıklı konuya bakın.

Ad çözünürlüğü

MQOPER çağrısı, kuyruğu ve kuyruk yöneticisi adlarını nasıl çözer.

Not: Kuyruk yöneticisi diğer adı, RNAME alanı olmayan bir uzak kuyruk tanımlamasıdır.

Bir WebSphere MQ kuyruğu açtığınızda, MQOPEN çağrısı belirttiğiniz kuyruk adında bir ad çözme işlevi gerçekleştirir. Bu, kuyruk yöneticisinin sonraki işlemleri hangi kuyruğa gerçekleştireceğini belirler. Diğer bir deyişle, bir diğer ad kuyruğunun adını ya da nesne tanımlayıcınızda (MQOD) uzak bir kuyruk belirlediğinizde, çağrı adı yerel bir kuyrukta ya da bir iletim kuyruğuna çözer. Herhangi bir giriş, göz atma ya da küme için bir kuyruk açılırsa, bu bir kuyruk varsa, yerel bir kuyruğa çözülür ve bir hata varsa, bu bir yerel kuyruğa çözülür. Yalnızca çıkış için açılmışsa, yalnızca çıkış için açılmışsa, yalnızca yalnızca sorguların ya da çıkışın ve sorgu için açılırsa yerel olmayan bir kuyruğa çözülür. Ad çözme işlemine ilişkin genel bilgiler için bkz. Çizelge 34 sayfa 208 . *ObjectQMgrName* içinde sağladığınız ad, *ObjectName*' ta *önce* çözümlenir.

Çizelge 34 sayfa 208 ayrıca, kuyruk yöneticisi adı için bir diğer ad tanımlamak üzere uzak bir kuyruğun yerel tanımlamasını nasıl kullanabildiğinizi de gösterir. Bu, iletileri uzak bir kuyruğa koyduğunuzda hangi iletim kuyruğunun kullanılacağını seçmenizi sağlar. Böylece, uzak kuyruk yöneticilerine gönderilen iletiler için tek bir iletim kuyruğu kullanabilirsiniz.

Aşağıdaki çizelgeyi kullanmak için, önce soldaki iki kolonu (**MQOD ' ye giriş**başlığı altında) okuyun ve uygun vakayı seçin. Ardından, yönergeleri izleyerek karşılık gelen satırı okuyun. **Çözülen adlar** kolonlarındaki yönergeleri izleyerek, **MQOD ' ye giriş** kolonlarına geri dönebilir ve belirtildiği gibi değerleri ekleyebilirsiniz; tersi durumda, belirtilen sonuçlarla çizelgeden çıkabilirsiniz. Örneğin, *ObjectName*girişini yapmak zorunda kalabilirsiniz.

Çizelge 34. MQOPEN kullanılırken kuyruk adlarının çözülmesi

MQOD ' ye giriş		**Çözümlenen adlar**		
ObjectQMgrName	***ObjectName***	***ObjectQMgrName***	***ObjectName***	**İletim kuyruğu**
Boş ya da yerel kuyruk yöneticisi	CLUSTER özniteliği olmayan yerel kuyruk	Yerel kuyruk yöneticisi	Giriş *ObjectName*	Geçerli değil (yerel kuyruk kullanıldı)
Boş kuyruk yöneticisi	CLUSTER özniteliği olan yerel kuyruk	İş yükü yönetimi seçilen küme kuyruk yöneticisi ya da PUT üzerinde seçilen belirli bir küme kuyruk yöneticisi	Giriş *ObjectName*	SYSTEM.CLUSTER.TRANSMIT.QUEUE ve yerel kuyruk kullanıldı SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Yerel kuyruk yöneticisi	CLUSTER özniteliği olan yerel kuyruk	Yerel kuyruk yöneticisi	Giriş *ObjectName*	Geçerli değil (yerel kuyruk kullanıldı)
Boş ya da yerel kuyruk yöneticisi	Model kuyruğu	Yerel kuyruk yöneticisi	Oluşturulan ad	Geçerli değil (yerel kuyruk kullanıldı)

Çizelge 34. MQOPEN kullanılırken kuyruk adlarının çözülmesi (devamı var)

MQOD ' ye giriş		**Çözümlenen adlar**		
ObjectQMgrName	***ObjectName***	***ObjectQMgrName***	***ObjectName***	**İletim kuyruğu**
Boş ya da yerel kuyruk yöneticisi	CLUSTER özniteliği olan ya da olmayan diğer ad kuyruğu	Perform name resolution again with *ObjectQMgrAd* unchanged, and input *ObjectName* set to the *BaseQName* in the alias queue definition object. *ObjectQMgrAdı* belirtildiğinde yerel olarak tanımlanmış bir diğer ada çözümlenmemelidir, ancak *ObjectQMgrName* ' in boş olduğu, kümelenmiş diğer ada (diğer kuyruk yöneticilerine ev sahipliği yaptı) çözümlenmelidir.		
Yerel kuyruk yöneticisi	CLUSTER özniteliği olan diğer ad kuyruğu	Diğer ad, yerel olarak tanımlanmış olmayan bir küme kuyruğuna ya da diğer ad ile aynı *ObjectName* olan bir küme kuyruğuna çözümlenmemelidir.		
Boş kuyruk yöneticisi	CLUSTER özniteliği olan diğer ad kuyruğu	Diğer ad, diğer ad olarak aynı *ObjectName* olan bir küme kuyruğuna çözülabilir.		
Boş ya da yerel kuyruk yöneticisi	Uzak kuyruğun yerel tanımlaması	*ObjectQMgrAd* kümesi *RemoteQMgrAd*olarak ve *ObjectName* *RemoteQName*olarak ayarlanmış şekilde ad çözümlemesini yeniden gerçekleştirin. Uzak kuyruklar çözülmemelidir		*XmitQName* özniteliğinin adı, blank; değilse, uzak kuyruk tanımlaması nesnesindeki *RemoteQMgrName* (RemoteQMgr) adı. SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Boş kuyruk yöneticisi	Eşleşen yerel nesne yok; küme kuyruğu bulundu	İş yükü yönetimi seçilen küme kuyruk yöneticisi ya da PUT üzerinde seçilen belirli bir küme kuyruk yöneticisi	Giriş *ObjectName*	SYSTEM.CLUSTER.TRANSMIT.QUEUE SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Boş ya da yerel kuyruk yöneticisi	Eşleşen yerel nesne yok; küme kuyruğu bulunamadı		Hata, kuyruk bulunamadı	Burada geçerli değil

Çizelge 34. MQOPEN kullanılırken kuyruk adlarının çözülmesi (devamı var)

MQOD ' ye giriş		**Çözümlenen adlar**		
ObjectQMgrName	***ObjectName***	***ObjectQMgrName***	***ObjectName***	**İletim kuyruğu**
Kuyruk yöneticisinin yerel kuyruk yöneticisi olarak aynı kuyruk paylaşım grubunda yer alan adı	Yerel paylaşılan kuyruk	Yerel kuyruk yöneticisi	Giriş *ObjectName*	Burada geçerli değil
Yerel iletim kuyruğunun adı	(Çözümlenmedi)	Giriş *ObjectQMgrAdı*	Giriş *ObjectName*	Giriş *ObjectQMgrAdı* SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Kuyruk yöneticisi diğer adı tanımlaması (yerel kuyruk yöneticisi*RemoteQMgrAd* olabilir)	(Çözümlenmedi, uzak kuyruk)	Perform name resolution again with *ObjectQMgrAd* set to *RemoteQMgrAd*. Uzak kuyruklara çözümlenmemelidir	Giriş *ObjectName*	*XmitQName* özniteliğinin adı, blank; değilse, uzak kuyruk tanımlaması nesnesindeki *RemoteQMgrName* (RemoteQMgr) adı. SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Kuyruk yöneticisi herhangi bir yerel nesnenin adı değil; küme kuyruğu yöneticileri ya da kuyruk yöneticisi diğer adı bulundu	(Çözümlenmedi)	*ObjectQMgrAd* or specific cluster queue manager selected on PUT	Giriş *ObjectName*	SYSTEM.CLUSTER.TRANSMIT.QUEUE SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)
Kuyruk yöneticisi herhangi bir yerel nesnenin adı değil; herhangi bir küme nesnesi bulunamadı	(Çözümlenmedi)	Giriş *ObjectQMgrAdı*	Giriş *ObjectName*	*DefXmitQName* attribute of the queue manager where *DefXmitQName* is supported. SYSTEM.QSG.TRANSMIT.QUEUE (bkz. not)

Notlar:

1. *BaseQName* , diğer ad kuyruğunun tanımlamasından temel kuyruğun adıdır.
2. *RemoteQName* , uzak kuyruğun yerel tanımlamasından uzak kuyruğun adıdır.
3. *RemoteQMgrName* , uzak kuyruğun yerel tanımlamasından uzak kuyruk yöneticisinin adıdır.
4. *XmitQName* , uzak kuyruğun yerel tanımlamasından iletim kuyruğunun adıdır.
5. Bir kuyruk paylaşım grubunun (QSG) bir parçası olan z/OS kuyruk yöneticileri için WebSphere MQ kullanırken, Çizelge 34 sayfa 208içinde yerel kuyruk yöneticisi adı yerine QSG adı kullanılabilir.

   Yerel kuyruk yöneticisi hedef kuyruğu açamıyorsa ya da kuyruğa bir ileti yerleştiremezse, ileti, belirtilen ObjectQMgradına, grup içi kuyruklama ya da WebSphere MQ kanalı yoluyla aktarılır.
6. Çizelgenin *ObjectName* kolonunda CLUSTER, kuyruğun hem CLUSTER, hem de CLUSNL özniteliklerine gönderme yapar.
7. SYSTEM.QSG.TRANSMIT.QUEUE değeri, yerel ve uzak kuyruk yöneticileri aynı kuyruk paylaşım grubunda yer alıyorsa, grup içi kuyruklama geçerli kılınmışsa kullanılır.

8. Her bir küme gönderici kanalına farklı bir küme iletim kuyruğu atadıysanız, SYSTEM.CLUSTER.TRANSMIT.QUEUE , küme iletim kuyruğunun adı olmayabilir. Birden çok küme iletim kuyruğuna ilişkin ek bilgi için Clustering: Planning how to configure cluster transmising queus başlıklı konuya bakın.
9. Kuyruk yöneticisinin herhangi bir yerel nesnenin adı olmadığı durumlarda; küme kuyruğu yöneticileri ya da bulunan kuyruk yöneticisi diğer adı bulunur.

   ObjectQMgrNamekomutunu kullanarak bir kuyruk yöneticisi adı sağladığınızda ve yerel kuyruk yöneticisi tarafından bu hedefe ulaşabilecek farklı küme adları olan birden çok küme kanalı varsa, hedef kuyruğun küme adıyla bağımsız olarak, iletiyi taşımak için bu kanallardan herhangi biri kullanılabilir.

   Yalnızca kuyruğun aynı küme adına sahip bir kanaldan gönderilmek üzere o kuyruğa ilişkin iletileri bekliyorsanız, bu beklenmeyen bir durum olabilir.

   Ancak, **ObjectQMgrName** bu durumda öncelikli olur ve küme iş yükü dengelemesi, içinde oldukları küme adı ne olursa olsun, o kuyruk yöneticisine ulaşabilecek tüm kanalları dikkate alır.

Bir diğer ad kuyruğunun açılması, diğer adın çözdüğü temel kuyruğu da açar ve uzak bir kuyruk açılırsa iletim kuyruğunu da açar. Bu nedenle, belirttiğiniz kuyruğu ya da diğerinin açık olduğu kuyrukları silemez.

Bir diğer ad kuyruğu yerel olarak tanımlanmış başka bir diğer ad kuyruğuna (bir kümede paylaşılan ya da değil) çözümlenemediğinde, uzaktan tanımlanan bir diğer ad kuyruğunun çözülmesi izin verilir ve bu nedenle temel kuyruk olarak belirlenebilir.

Çözümlenen kuyruk adı ve çözülen kuyruk yöneticisi adı, MQOD ' daki *ResolvedQName* ve *ResolvedQMgrName* alanlarında saklanır.

Dağıtılmış bir kuyruğa alma ortamındaki ad çözümlemesi hakkında daha fazla bilgi için Kuyruk adı çözümlemesi nedir?başlıklı konuya bakın.

MQOPER çağrısının seçeneklerinin kullanılması

MQOPER çağrısının *Options* değiştirgesinde, açtığınız nesneye verdiğiniz erişimi denetlemek için bir ya da daha çok seçenek seçmeniz gerekir. Bu seçeneklerle şunları yapabilirsiniz:

- Bir kuyruğu açın ve o kuyruğa koyulan tüm iletilerin, aynı yönetim ortamına yönlendirilmesi gerektiğini belirtin
- İleti koymanıza izin vermek için bir kuyruk açın.
- İletilere göz atmanızı sağlamak için bir kuyruk açın
- Kuyruktan ileti kaldırmanıza izin vermek için bir kuyruk açın
- Özniteliklerini sorgulamanıza ve ayarlamasına izin vermek için bir nesneyi açın (ancak yalnızca kuyrukların özniteliklerini ayarlayabilirsiniz)
- İletileri yayınlamak için bir konuyu ya da konu dizesini açın
- Bağlam bilgilerini bir iletiyle ilişkilendirin
- Güvenlik denetimleri için kullanılacak alternatif bir kullanıcı kimliği gösterir.
- Kuyruk yöneticisi susturulmuş durumdaysa, çağrıyı denetleyebilirsiniz.

Küme kuyruğu için MQOPEN seçeneği
Kuyruk tanıtıcısı için kullanılan bağ tanımı, MQBND_BIND_ON_OPEN, MQBND_BIND_NOT_FIXEDya da MQBND_BIND_ON_GROUPdeğerini alabilen *DefBind* kuyruk özniteliğinden alınır.

MQPUT komutunu kullanarak kuyruğa konan tüm iletileri aynı rotayla aynı kuyruk yöneticisine yöneltmek için MQOPEN çağrısında MQOO_BIND_ON_OPEN seçeneğini kullanın.

Bir hedefin MQPUT zamanında seçileceğini belirtmek için, yani ileti temelinde MQOPEN çağrısında MQOO_BIND_NOT_FIXED seçeneğini kullanın.

MQPUT kullanılarak kuyruğa konan ileti gruplarındaki tüm iletilerin aynı hedef örneğe ayrıldığını belirtmek için, MQOPEN çağrısında MQOO_BIND_ON_GROUP seçeneğini kullanın.

Gruptaki tüm iletilerin aynı hedefte işlendiğinden emin olmak için kümelerle ileti grupları kullanılırken MQOO_BIND_ON_OPEN ya da MQOO_BIND_ON_GROUP belirtilmelidir.

Bu seçeneklerden herhangi birini belirtmezseniz, MQOO_BIND_AS_Q_DEFvarsayılan değeri kullanılır.

MQODiçinde bir kuyruk yöneticisinin adını belirtirseniz, o kuyruk yöneticisindeki kuyruk seçilir. Kuyruk yöneticisi adı boşsa, herhangi bir eşgörünüm seçilebilir. Daha fazla bilgi için bkz. “MQOPEN ve kümeler” sayfa 331 .

Bir küme kuyruğunu QALIAS tanımı kullanarak açarsanız, bazı kuyruk öznitelikleri temel kuyruk değil, diğer ad kuyruğu tarafından tanımlanır. Küme öznitelikleri, diğer ad kuyruğu tarafından geçersiz kılınan temel kuyruk tanımlamasının öznitelikleri arasındadır. Örneğin, aşağıdaki parçacıkta, küme kuyruğu MQOO_BIND_ON_OPENile değil, MQOO_BIND_NOT_FIXED ile açılır. Küme kuyruğu tanımlaması kümede duyurulmuştur; diğer ad kuyruğu tanımı kuyruk yöneticisi için yereldir.

```
DEFINE QLOCAL(CLQ1)  CLUSTER(MYCLUSTER) DEFBIND(OPEN) REPLACE
DEFINE QALIAS(ACLQ1) TARGET(CLQ1) DEFBIND(NOTFIXED) REPLACE
```

İletileri koymak için MQOPEN seçeneği
İleti koymak üzere bir kuyruk ya da konu açmak için, MQOO_OUTPUT seçeneğini kullanın.

İletilere göz atmak için MQOPEN seçeneği
Bir kuyruğu açmak için, ilgili iletileri *göz atabilmeniz* için, MQOO_BROWSE seçeneğiyle MQOPEN çağrısını kullanın.

Bu, kuyruk yöneticisinin kuyruğun sonraki iletisini tanımlamak için kullandığı bir *göz atma imleçini* yaratır. Daha fazla bilgi için “Kuyruklardaki Iletilere Göz Atma” sayfa 260 başlıklı konuya bakın.

Not:

1. Uzak kuyruklardaki iletilere göz atamazsınız; MQOO_BROWSE seçeneğini kullanarak uzak kuyruk açmayın.
2. Bir dağıtım listesi açılırken bu seçeneği belirleyemezsiniz. Dağıtım listeleriyle ilgili daha fazla bilgi için bkz. “Dağıtım listeleri” sayfa 225.
3. İşbirliğine göz atma kullanıyorsanız, MQOO_CO_OP ' ı MQOO_BROWSE ile birlikte kullanın; bkz. Seçenekler

İletileri kaldırmak için MQOPEN seçenekleri
Üç seçenek, iletileri bu kuyruktan kaldırmak için bir kuyruğun açılmasını denetler.

Bu çizelgelerden yalnızca birini herhangi bir MQOPER çağrısında kullanabilirsiniz. Bu seçenekler, programınızın kuyruğa özel ya da paylaşılan erişim olup olmadığını tanımlar. *Dışlayıcı erişim* , kuyruğu kapatıncaya kadar,Yalnızca siz iletileri bu iletiden kaldırabilirsiniz. İletileri kaldırmak için başka bir program kuyruğu açma girişiminde bulunursa, MQOPEN çağrısı başarısız olur. *Paylaşılan erişim* , birden çok programın kaldırılabildiği anlamına gelirkuyruktan iletilere bakın.

En çok önerilebilen yaklaşım, kuyruk tanımlandığında kuyruk için tasarlanan erişim tipini kabul etmesidir. Kuyruk tanımı, *Shareability* ve/*DefInputOpenOption* öznitelikleri. Bu erişimi kabul etmek için, MQOO_INPUT_AS_Q_DEF seçeneğini kullanın. Bu özniteliklerin ayarının, bu seçeneği kullanırken size verilecek erişim tipini nasıl etkilediğini görmek için Çizelge 35 sayfa 212 dosyasına bakın.

Çizelge 35. MQOPER çağrısının kuyruk öznitelikleri ve seçenekleri, kuyruklara erişimi nasıl etkiler?

Kuyruk öznitelikleri		**MQREOL seçeneklerine ilişkin erişim tipi**		
Shareability	*DefInputOpenOption*	**AS_Q_DEF**	**Paylaşılan**	**dışlayıcı**
PAYLAşıM	Paylaşılan	paylaşılan	paylaşılan	dışlayıcı
PAYLAşıM	dışlayıcı	dışlayıcı	paylaşılan	dışlayıcı
NOT_SHAREABLE*	SHARED*	dışlayıcı	dışlayıcı	dışlayıcı
NOT_SHAREABLE	dışlayıcı	dışlayıcı	dışlayıcı	dışlayıcı

Çizelge 35. MQOPER çağrısının kuyruk öznitelikleri ve seçenekleri, kuyruklara erişimi nasıl etkiler? (devamı var)	
Kuyruk öznitelikleri	**MQREOL seçeneklerine ilişkin erişim tipi**
Not: * Bu özniteliklerin birleşimine sahip olmak için bir kuyruk tanımlayabilmenize rağmen, varsayılan giriş açma seçeneği, paylaşılabilirlik öznitemesiyle geçersiz kılınır.	

Alternatif olarak:

- Başka programlar kuyruktan aynı anda iletileri kaldırabilse bile uygulamanızın başarılı bir şekilde çalışabileceğini biliyorsanız, MQOO_INPUT_SHARED seçeneğini kullanın. Çizelge 35 sayfa 212 , bu seçenekle birlikte, bazı durumlarda kuyruğa dışlayıcı erişim verileceğini gösterir.
- Uygulamanızın, iletileri aynı anda kuyruktan kaldırmak için başka programlar önleniyorsa başarılı bir şekilde çalışabileceğini biliyorsanız, MQOO_INPUT_EXCLUSIVE seçeneğini kullanın.

Not:

1. İletileri uzak bir kuyruktan kaldıramazsınız. Bu nedenle, MQOO_INPUT_ * seçeneklerinden herhangi birini kullanarak uzak bir kuyruk açamazsınız.
2. Bir dağıtım listesi açılırken bu seçeneği belirleyemezsiniz. Daha fazla bilgi için bkz. “Dağıtım listeleri” sayfa 225.

Özniteliklerin ayarlanması ve araştırılmasına ilişkin MQOPEN seçenekleri
Bir kuyruğu açmak için, bu kuyruğun özniteliklerini ayarlayabilmeniz için MQOO_SET seçeneğini kullanın.

Diğer herhangi bir nesne tipinin özniteliklerini ayarlayamazsınız (bkz. “Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305).

Özniteliklerini sorgulayabilmeniz için bir nesneyi açmak için MQOO_INCOVER seçeneğini kullanın.

Not: Bir dağıtım listesi açılırken bu seçeneği belirleyemezsiniz.

İleti bağlamına ilişkin MQAç seçenekleri
Bir iletiyi bir kuyruğa koyduğunuzda bağlam bilgilerini bir iletiyle ilişkilendirebilmek istiyorsanız, kuyruğu açtığınızda ileti bağlamı seçeneklerinden birini kullanmanız gerekir.

Seçenekler, iletiyi oluşturan *kullanıcıyla* ilişkili bağlam bilgilerini ve iletiyi oluşturan *uygulama* ile ilgili olan bağlam bilgilerini birbirinden ayırt etmenizi sağlar. Ayrıca, iletiyi kuyruğa koyduğunuzda bağlam bilgilerini ayarlamayı tercih edebilir ya da bağlamın başka bir kuyruk saptından otomatik olarak alınmasını tercih edebilirsiniz.

İlgili kavramlar
“İleti bağlamı” sayfa 37
İleti bağlamı bilgileri, iletiyi alan uygulamanın, iletiyi oluşturan iletiyi bulmasını sağlayan bir uygulamaya olanak tanır.

“Bağlam bilgilerini denetleme” sayfa 222
Bir iletiyi kuyruğa koymak için MQPUT ya da MQPUT1 çağrısını kullandığınızda, kuyruk yöneticisinin ileti tanımlayıcısına varsayılan bağlam bilgileri eklemesini belirleyebilirsiniz. Uygun yetki düzeyine sahip uygulamalar ek bağlam bilgileri ekleyebilir. Bağlam bilgilerini denetlemek için, MQPMO yapısındaki seçenekler alanını kullanabilirsiniz.

Diğer kullanıcı yetkisi için MQOPEN seçeneği
MQOPER çağrısını kullanarak bir nesneyi açmaya çalıştığınızda, kuyruk yöneticisi o nesneyi açma yetkisine sahip olup olmadığını denetler. Yetkiniz yoksa, arama başarısız olur.

Ancak, sunucu programları, kuyruk yöneticisinin, sunucunun kendi yetkisi yerine, üzerinde çalışmakta oldukları kullanıcının yetkilendirmesini denetmesini isteyebilirler. Bunu yapmak için, bu kullanıcılar, MQOPED çağrısının MQOO_ALTERNATE_USER_AUTHORITY seçeneğini kullanmalı ve MQOD yapısının *AlternateUserId* alanında diğer kullanıcı kimliğini belirtmelidir. Genellikle, sunucu, işlem yaptığı iletteki bağlam bilgilerinden kullanıcı kimliğini alır.

Kuyruk yöneticisi için MQOPEN seçeneği susturulmuş
z/OSüzerindeki CICS ortamında, kuyruk yöneticisi susturma durumundaysa, MQOPER çağrısını kullanırsanız, çağrı her zaman başarısız olur.

Diğer z/OS ortamlarında, IBM i, Windows sistemlerinde ve UNIX and Linux sistemleri ortamlarında, kuyruk yöneticisi yalnızca MQOPEN çağrısının MQOO_FAIL_IF_QUIESCING seçeneğini kullanıyorsanız, çağrı başarısız olur.

Yerel kuyruk adlarını çözmek için MQOPEN seçeneği
Yerel, diğer ad ya da model kuyruğunu açtığınızda, yerel kuyruk döndürülür.

Ancak, uzak bir kuyruğu ya da küme kuyruğunu açtığınızda, MQOD yapısının *ResolvedQName* ve *ResolvedQMgrName* alanları, uzak kuyruk tanımında bulunan uzak kuyruk ya da uzak kuyruk yöneticisi adlarıyla ya da seçilen uzak küme kuyruğunda doldurulur.

MQOP çağrısının MQOO_RESOLVE_LOCAL_Q seçeneğini kullanarak, açıldığı yerel kuyruğun adıyla MQOD yapısındaki *ResolvedQName* öğesini doldurmanız gerekir. *ResolvedQMgrName* , aynı şekilde yerel kuyruğu barındıran yerel kuyruk yöneticisinin adıyla doldurulur. Bu alan yalnızca MQOD yapısındaki Sürüm 3 ile kullanılabilir; yapı Sürüm 3 'ten küçükse, bir hata döndürülmeden MQOO_RESOLVE_LOCAL_Q yoksayılır.

If you specify MQOO_RESOLVE_LOCAL_Q when opening, for example, a remote queue, *ResolvedQName* is the name of the transmission queue to which messages will be put. *ResolvedQMgrName*, iletim kuyruğunu bulunduran yerel kuyruk yöneticisinin adıdır.

Dinamik kuyruklar oluşturma

Uygulamanızın sona ermesinden sonra kuyruğa gerek kalmadığında dinamik bir kuyruk kullanın.

Örneğin, yanıtlama kuyruğunuz için dinamik bir kuyruk kullanabilirsiniz. Bir kuyruğa ileti koyduğunuzda, MQMD yapısının *ReplyToQ* alanında yanıtlanacak yanıtın adını belirtmiş olun (bkz. “MQMD yapısıyla iletilerin tanımlanması” sayfa 217).

Dinamik bir kuyruk yaratmak için, MQOPEN çağrısıyla birlikte, model kuyruğu olarak bilinen bir şablonu kullanıyorsunuz. You create a model queue using the WebSphere MQ commands or the operations and control panels. Yarattığınız dinamik kuyruk, model kuyruğun özniteliklerini alır.

MQOL ' u çağırdığınızda, MQOD yapısının *ObjectName* alanında model kuyruğunun adını belirtin. Arama tamamlandığında, *ObjectName* alanı oluşturulan dinamik kuyruğun adına ayarlanır. Ayrıca, *ObjectQMgrName* alanı yerel kuyruk yöneticisinin adına ayarlıdır.

Yarattığınız dinamik kuyruğun adını üç yolla belirtebilirsiniz:

- MQOD yapısındaki *DynamicQName* alanında istediğiniz tam adı verin.
- Ad için bir önek (en az 33 karakter) belirtin ve kuyruk yöneticisinin adın geri kalanını oluşturmasını sağlayın. Bu, kuyruk yöneticisinin benzersiz bir ad oluşturduğu, ancak yine de bazı denetime sahip olduğunuz anlamına gelir (örneğin, her kullanıcının belirli bir öneki kullanmasını ya da adlarında belirli bir öneke sahip kuyruklar için özel bir güvenlik sınıflandırması vermek isteyebilirsiniz). Bu yöntemi kullanmak için, *DynamicQName* alanının en son olmayan karakteri için bir yıldız işareti (*) belirtin. Dinamik kuyruk adı için tek bir yıldız imi (*) belirlemeyin.
- Kuyruk yöneticisinin tam adı oluşturmasına izin verin. Bu yöntemi kullanmak için, *DynamicQName* alanının ilk karakter konumunda bir yıldız işareti (*) belirtin.

Bu yöntemlere ilişkin ek bilgi için DynamicQName (Dinamik QName) alanının tanımına bakın.

Dinamik ve Model kuyrukları alanında dinamik kuyruklara ilişkin daha fazla bilgi vardır.

Uzak Kuyrukların Açılması

Uzak kuyruk, uygulamanın bağlı olduğu kuyruk yöneticisinin sahip olduğu bir kuyruktır.

Uzak bir kuyruğu açmak için, MQOPEN çağrısını yerel bir kuyruk olarak kullanın. Kuyruğun adını aşağıdaki gibi belirtebilirsiniz:

1. MQOD yapısının *ObjectName* alanında, uzak kuyruğun adını *yerel* kuyruk yöneticisi tarafından bilindiği şekilde belirtin.

   Not: Bu durumda *ObjectQMgrName* alanını boş bırakın.
2. MQOD yapısının *ObjectName* alanında, *uzak* kuyruk yöneticisinde bilindiği gibi uzak kuyruğun adını belirtin. *ObjectQMgrName* alanında aşağıdakilerden birini belirtin:
   - Uzak kuyruk yöneticisiniyle aynı adı taşıyan iletim kuyruğunun adı. Ad ve büyük/küçük harf (büyük harf, küçük harf ya da bir karışım) *tam olarak*eşleşmelidir.
   - Hedef kuyruk yöneticisine ya da iletim kuyruğuna çözülen, kuyruk yöneticisi diğer adı nesnesinin adı.

   Bu, kuyruğun yöneticisine iletinin yerini ve oraya ulaşmak için gereken iletim kuyruğunu belirtir.
3. *DefXmitQname* destekleniyorsa, MQOD yapısının *ObjectName* alanında, *remote* kuyruk yöneticisi tarafından bilinen uzak kuyruğun adını belirtin.

   Not: *ObjectQMgrName* alanını uzak kuyruk yöneticisinin adına ayarlayın (bu durumda boş bırakılamaz).

MQOL ' u çağırdığınızda yalnızca yerel adların geçerliliği denetlenir; son denetim, kullanılacak iletim kuyruğunun varolması içindir.

Bu yöntemler Çizelge 34 sayfa 208içinde özetlenmiştir.

MQCLOSE çağrısını kullanarak nesneler kapatılıyor

Bir nesneyi kapatmak için, MQCLOSE çağrısını kullanın.

Nesne kuyruksa, aşağıdakine dikkat edin:

- Kapatmadan önce geçici bir dinamik kuyruğu boşaltmanıza gerek yoktur.

  Geçici bir dinamik kuyruğu kapattığınızda, kuyruğun üzerinde hala üzerinde olabilecek iletilerle birlikte kuyruk silinir. Kuyruğa karşı işlenmemiş MQGET, MQPUT ya da MQPUT1 çağrıları varsa bu doğru olur.
- z/OSiçin WebSphere MQ ' ta, o kuyruk için MQGMO_SET_SIGNAL seçeneği bekleyen bir MQGET isteğiniz varsa, bunlar iptal edilir.
- Kuyruğu MQOO_BROOK seçeneğini kullanarak açdıysanız, göz atma imleçiniz yok edilir.

Kapanış, eşitleme noktalarıyla ilgisiz olduğundan, eşitleme noktasından önce ya da sonra kuyrukları kapatabilirsiniz.

MQCLOSE çağrısına giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı. Bunu açmak için kullanılan bağlantı tanıtıcısını kullanın ya da diğer bir seçenek olarak, z/OSüzerindeki CICS uygulamaları için, değişmez MQHC_DEF_HCONN (sıfır değerine sahip) değerini belirtebilirsiniz.
- Kapatmak istediğiniz nesnenin saptanıdır. Bu işlemi, MQOPEN çağrısının çıkışından alın.
- *Options* alanında MQCO_NONE (kalıcı bir dinamik kuyruğu kapatmadığınız sürece).
- Kuyruk yöneticisinin, üzerinde hala ileti olup olmadığını (kalıcı bir dinamik kuyruğu kapattığında) bile silmesi gerekip gerekmediğini belirlemek için denetim seçeneği.

MQCLOSE komutunun çıkışı şöyledir:

- Tamamlanma kodu
- Neden kodu
- Nesne tanıtıcısı, MQHO_UNUSABLE_HOBJ değerini ilk durumuna getirir

Descriptions of the parameters of the MQCLOSE call are given in MQCLOSE.

İletileri Kuyruğa Koyma

İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

İletileri kuyruğa koymak için MQPUT çağrısını kullanın. MQPUT ' u, ilk MQOPEN çağrısını izleyerek, aynı kuyruğa birden çok ileti koymak için sık olarak kullanabilirsiniz. Tüm iletilerinizi kuyruğa koyduğunuzda MQCLOSE ' yi çağırın.

Bir kuyruğa tek bir ileti koymak ve kuyruğu hemen sonra kapatmak isterseniz, MQPUT1 çağrısını kullanabilirsiniz. MQPUT1 , aşağıdaki çağrılar sırasıyla aynı işlevleri gerçekleştirir:

- MQOPEN
- MQPUT
- MQCLOSE

Genellikle, kuyruğa koymak için birden çok iletiniz varsa, MQPUT çağrısını kullanmak daha verimli olur. Bu, iletinin boyutuna ve üzerinde çalışmakta olduğunuz altyapıya bağlıdır.

Bir kuyruğa ileti yerleştirilmesiyle ilgili daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “MQPUT çağrısını kullanarak iletileri yerel bir kuyruğa koyma” sayfa 216
- “İletileri Uzak Kuyruğa Koyma” sayfa 221
- “İletiye ilişkin özelliklerin ayarlanması” sayfa 221
- “Bağlam bilgilerini denetleme” sayfa 222
- “MQPUT1 çağrısını kullanarak bir ileti kuyruğa konması” sayfa 224
- “Dağıtım listeleri” sayfa 225
- “Put çağrılarının başarısız olduğu bazı durumlar” sayfa 229

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

MQPUT çağrısını kullanarak iletileri yerel bir kuyruğa koyma

MQPUT çağrısını kullanarak iletileri yerel bir kuyruğa koyma hakkında bilgi edinmek için bu bilgileri kullanın.

MQPUT çağrısına giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bir bağlantı tanıtıcısı (Hconn).
- Bir kuyruk tanıtıcısı (Hobj).

- Kuyruğa koymak istediğiniz iletinin açıklaması. Bu, ileti tanımlayıcı yapısı (MQMD) biçimidir.
- Denetim bilgileri, bir put-message options yapısı (MQPMO) biçiminde olur.
- İleti içinde yer alan verilerin uzunluğu (MQlong).
- İleti verilerinin kendisi.

MQPUT çağrısından çıkış aşağıdaki gibidir:

- Neden kodu (MQUBE)
- Tamamlanma kodu (MQUBE)

Arama başarıyla tamamlanırsa, seçenek yapılarınızı ve ileti tanımlayıcı yapınızı da döndürür. Çağrı, seçenek yapınızı, iletinin adını ve iletinin gönderildiği kuyruk yöneticisini gösterecek şekilde değiştirir. Kuyruk yöneticisinin koymakta olduğunuz iletinin tanıtıcısı için benzersiz bir değer oluşturduğunu (MQMD yapısının *MsgId* alanında ikili sıfır belirterek) istediyseniz, çağrı size bu yapıyı döndürmeden önce *MsgId* alanına değeri ekler. Başka bir MQPUT yayınlamadan önce bu değeri ilk durumuna getirin.

MQPUTiçinde MQPUT çağrısının bir açıklaması var.

MQPUT çağrısına giriş olarak gereken bilgilere ilişkin ek bilgi için aşağıdaki bağlantılara bakın:

- “Tanıtıcıların belirtilmesi” sayfa 217
- “MQMD yapısıyla iletilerin tanımlanması” sayfa 217
- “MQPMO yapısını kullanarak seçenekleri belirtme” sayfa 217
- “İletinizdeki veriler” sayfa 220
- “İleti koyma: İleti tanıtıcılarını kullanma” sayfa 221

Tanıtıcıların belirtilmesi

z/OS uygulamalarında CICS içindeki bağlantı tanıtıcısı (*Hconn*) için, değişmez MQHC_DEF_HCONN değerini (sıfır değerine sahip) belirleyebilir ya da MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanabilirsiniz. Diğer uygulamalar için, her zaman MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanın.

Çalışmakta olduğunuz ortam ne olursa olsun, MQOPEN çağrısının döndürdüğü kuyruk tanıtıcısını (*Hobj*) kullanın.

MQMD yapısıyla iletilerin tanımlanması

İleti tanımlayıcı yapısı (MQMD), MQPUT ve MQPUT1 çağrılarına ilişkin bir giriş/çıkış değiştirgedir. Kuyruğa koymakta olduğunuz iletiyi tanımlamak için bu programı kullanın.

İleti için MQPRI_PRIORITY_AS_Q_DEF ya da MQPER_PERSISTENCE_AS_Q_DEF değeri belirtilirse ve kuyruk bir küme kuyruğalıysa, kullanılan değerler MQPUT ' un çözdüğü kuyruklardır. Bu kuyruk MQPUT için geçersiz kılındıysa, çağrı başarısız olur. Ek bilgi için Kuyruk yöneticisi kümesinin yapılandırılması başlıklı konuya bakın.

Not: *MsgId* ve *CorrelId* 'in benzersiz olduğundan emin olmak için yeni bir ileti koymadan önce MQPMO_NEW_MSG_ID ve MQPMO_NEW_CORREL_ID' yi kullanın. Bu alanlardaki değerler başarılı bir MQPUT ' un (MQPUT) içinde döndürülür.

MQMD 'nin “IBM WebSphere MQ ileti” sayfa 9' de tanımladığı ve MQMD' ta yapısının bir açıklaması olan ileti özelliklerine ilişkin bir giriş vardır.

MQPMO yapısını kullanarak seçenekleri belirtme

MQPUT ve MQPUT1 çağrılarına seçenekler geçirmek için MQPMO (İleti Koyma Seçeneği Ekle) yapısını kullanın.

Aşağıdaki kısımlar, bu yapının alanlarını doldurmada size yardımcı olur. There is a description of the structure in MQPMO.

Yapı, aşağıdaki alanları içerir:

- *StrucId*
- *Version*
- *Options*
- *Context*
- *ResolvedQName*
- *ResolvedQMgrName*
- *RecsPresent*
- *PutMsgRecsFields*
- *ResponseRecOffset and ResponseRecPtr*
- *OriginalMsgHandle*
- *NewMsgHandle*
- *Action*
- *PubLevel*

Bu alanların içeriği aşağıdaki gibidir:

StrucId
Bu, yapıyı put-message options yapısı olarak tanımlar. Bu, 4 karakterlik bir alandır. Her zaman MQPMO_STRUC_ID değerini belirtin.

S\u00fcr\u00fcm
Bu, yapının sürüm numarasını açıklar. Varsayılan değer MQPMO_VERSION_1' dir. MQPMO_VERSION_2yazarsanız, dağıtım listelerini kullanabilirsiniz (bkz. “Dağıtım listeleri” sayfa 225). MQPMO_VERSION_3değerini girerseniz, ileti çekme noktaları ve ileti özelliklerini kullanabilirsiniz. MQPMO_CURRENT_VERSION değerini girerseniz, uygulamanız her zaman en son düzeyi kullanacak şekilde ayarlanır.

Seçenekler
Bu denetim, aşağıdakileri denetler:

- Put işleminin bir iş birimine dahil edilip edilip etmeyeceğini
- Bir iletiyle ne kadar bağlam bilgisi ilişkilendirilir?
- Bağlam bilgilerinin alındığı yer
- Kuyruk yöneticisi susturulmuş durumda olduğunda çağrıların başarısız olup olmadığını
- Gruplamaya ya da bölümlemeye izin verilip
- Yeni ileti tanıtıcısı ve ilinti tanıtıcısı oluşturma
- İletilerin ve bölümlerin bir kuyruğa konacağı sıralama düzeni
- Yerel kuyruk adlarının çözümlenip çözümlenmeyeceği

Options alanını varsayılan değere (MQPMO_NONE) bırakırsanız, koydugunuz ileti, onunla ilişkili varsayılan bağlam bilgilerini içerir.

Ayrıca, arama, eşitleme noktalarıyla çalışma biçiminin altyapıya göre belirlenmesine yol gösterir. Eşitleme noktası denetimi varsayılan değeri, z/OS' ta evet; diğer platformlar için ise hayır.

Bağlam
Bu durum, (*Options* alanında istenirse), bağlam bilgilerinin kopyalanacağı kuyruk tanıtıcısı adını belirtir.

İleti bağlamına ilişkin bir giriş için bkz. “İleti bağlamı” sayfa 37. Bir iletide bağlam bilgilerini denetlemek için MQPMO yapısının kullanılmasıyla ilgili bilgi edinmek için bkz. “Bağlam bilgilerini denetleme” sayfa 222.

ResolvedQName
Bu, iletiyi almak için açılan kuyruğun adını (diğer ad adını çözümledikten sonra) içerir. Bu bir çıkış alanıdır.

ResolvedQMgrAdı
Bu, *ResolvedQName*içinde kuyruğa sahip olan kuyruk yöneticisinin adını (diğer ad adını çözümledikten sonra) içerir. Bu bir çıkış alanıdır.

MQPMO, dağıtım listeleri için gereken alanları da barındırabilir (bkz. “Dağıtım listeleri” sayfa 225). Bu olanağı kullanmak istiyorsanız, MQPMO yapısının Sürüm 2 kullanılır. Bu, aşağıdaki alanları içerir:

RecsPresent
Bu alan, dağıtım listesindeki kuyrukların sayısını, yani MQPMR (Put Message Records) ve karşılık gelen Yanıt Kayıtların (MQRR) sayısını içerir.

Girdiğiniz değer, MQOPEN ' da sağlanan Nesne Kayıtları sayısıyla aynı olabilir. Ancak, değer MQOPEN çağrısında sağlanan Nesne Kaydı sayısından azsa ya da herhangi bir Ileti Kaydı Koyma değeri sağlıyorsa, tanımlanmamış olan kuyrukların değerleri, ileti tanımlayıcısı tarafından sağlanan varsayılan değerlerden alınır. Ayrıca, değer, sağlanan Nesne Kayıtları sayısından büyükse, Fazla Olan İleti Kayıtları dikkate alınmaz.

Aşağıdakilerden birini yapmanız için önerildiniz:

- Her hedeften bir rapor ya da yanıt almak istiyorsanız, MQOR yapısındaki gibi, aynı değeri girin ve *MsgId* alanlarını içeren MQPR ' leri kullanın. Bu *MsgId* alanlarını sıfırlarla başlatın ya da MQPMO_NEW_MSG_ID değerini belirtin.

  İletiyi kuyruğa koyduğunuzda, kuyruk yöneticisinin oluşturduğu *MsgId* değerleri, MQPMR; ' da kullanılabilir duruma gelir; bu değerleri, her bir raporla ya da yanıta ilişkin hedefi tanımlamak için kullanabilirsiniz.
- Rapor ya da yanıt almak istemezseniz, aşağıdakilerden birini seçin:
  1. If you want to identify destinations that fail immediately, you might still want to enter the same value in the *RecsPresent* field as appears in the MQOR structure and provide MQRRs to identify these destinations. Herhangi bir MQPMR belirtmeyin.
  2. Başarısız olan hedefleri tanımlamak istemiyorsanız, *RecsPresent* alanına sıfır girin ve MQPR 'leri ya da MQRR' leri sağlamayın.

Not: MQPUT1kullanıyorsanız, Yanıt Kaydı Göstergelerinin sayısı ve Yanıt Kaydı Offsets sayısı sıfır olmalıdır.

İleti Kayıtlarının (MQPMR) ve Yanıt Kayıtlarının (MQRR) tam açıklaması için MQPMR ve MQRRbaşlıklı konuya bakın.

PutMsgRecFields
Bu, her Put Message Record (MQPMR) ' da hangi alanların bulunduğunu belirtir. Bu alanların bir listesi için bkz. “MQPMR yapısının kullanılması” sayfa 229.

PutMsgRecOffset* ve *PutMsgRecPtr
Göstergeler (genellikle C) ve göreli konumlar (genellikle COBOL ' da) İleti Koyma Kayıtlarını çözmek için kullanılır (MQPMR yapısına genel bir bakış için bkz. “MQPMR yapısının kullanılması” sayfa 229).

İlk Put Message Record (İleti Kaydı) ya da ilk put iletisi kaydının göreli konumunu belirtmek için *PutMsgRecOffset* alanını kullanarak *PutMsgRecPtr* alanını kullanın. Bu, MQPMO ' nun başlangıcından göreli konudur. *PutMsgRecFields* alanına bağlı olarak, *PutMsgRecOffset* ya da *PutMsgRecPtr*için boş olmayan bir değer girin.

ResponseRecGöreli Konumu ve ResponseRecPtr
Ayrıca, Yanıt Kayıtlarına yanıt vermek için işaretçiler ve göreli konumlar da kullanıyorsunuz (Yanıt Kayıtları ile ilgili daha fazla bilgi için bkz. “MQRR yapısının kullanılması” sayfa 228).

İlk Yanıt Kaydına ilişkin bir gösterge belirlemek için *ResponseRecPtr* alanını ya da ilk Yanıt Kaydının göreli konumunu belirtmek için *ResponseRecOffset* alanını kullanın. Bu, MQPMO yapısının başlangıcından göreli konudur. *ResponseRecOffset* ya da *ResponseRecPtr*için boş olmayan bir değer girin.

Not: İletileri bir dağıtım listesine koymak için MQPUT1 kullanıyorsanız, *ResponseRecPtr* boş değerli ya da sıfır olmalı ve *ResponseRecOffset* değeri sıfır olmalıdır.

MQPMO yapısının 3. sürümü ek olarak aşağıdaki alanları da içerir:

OriginalMsgtanıtıcısı
Bu alanda yapabildiğiniz kullanım, *Eylem* alanının değerine bağlıdır. İlişkili ileti özellikleriyle yeni bir ileti yerleştiriyorsanız, bu alanı önceden yaratmış olduğunuz ileti tanıtıcısı olarak ayarlayın ve özellikleri bu alana ayarlayın. Önceden alınan bir iletiye yanıt olarak bir rapor iletiyorsanız, yanıtlıyorsanız ya da bir rapor oluşturuyorsanız, bu alan o iletinin ileti tanıtıcısını içerir.

NewMsgTanıtıcısı
If you specify a *NewMsgİşlemesi*, any properties associated with the handle override properties associated with the *OriginalMsgİşlemesi*. Daha fazla bilgi için bkz. Action (MQUZE).

İşlem
Gerçekleştirilmekte olan kont tipinin tipini belirtmek için bu alanı kullanın. Olası değerler ve anlamları aşağıdaki gibidir:

MQACTP_YENİ
Bu başka bir mesajla alakasız yeni bir mesaj.

MQAKP_ILERI
Bu ileti önceden alındı ve şimdi iletiliyor.

MQACTP_CEVAPLA
Bu ileti, önceden alınan bir iletinin yanıtına neden olur.

MQACP_REPORT
Bu ileti, önceden alınan bir iletinin sonucu olarak oluşturulan bir rapordur.

Daha fazla bilgi için bkz. Action (MQUZE).

PubLevel
Bu ileti bir yayınsa, hangi aboneliklerin hangi abonelikleri alacağını belirlemek için bu alanı ayarlayabilirsiniz. Yalnızca *SubLevel* değeri bu değerden küçük ya da bu değere eşit olan abonelikler bu yayını alır. Varsayılan değer, en yüksek düzey olan 9 'tır ve herhangi bir *SubLevel* ' e sahip aboneliklerin bu yayını alabileceği anlamına gelir.

İletinizdeki veriler

MQPUT çağrısının *Buffer* parametresindeki verinizi içeren arabelleğin adresini verin. İletilerinizdeki verilere herhangi bir şey dahil edebilirsiniz. Ancak iletilerde bulunan veri miktarı, bunları işleyen uygulamanın performansını etkiler.

Veri büyüklüğü üst sınırı aşağıdaki tarafından belirlenir:

- Kuyruk yöneticisinin *MaxMsgLength* özniteliği
- İletiyi koymakta olduğunuz kuyruğun *MaxMsgLength* özniteliği
- WebSphere MQ (ölü-harf üstbilgisi, MQDLH ve dağıtım listesi üstbilgisi, MQDH gibi) tarafından eklenen herhangi bir ileti üstbilgisi boyutu

Kuyruk yöneticisinin *MaxMsgLength* özniteliği, kuyruk yöneticisinin işleyebileceği ileti büyüklüğünü tutar. Bu, V6 ya da daha sonraki bir sürüm olan tüm WebSphere MQ ürünleri için 100 MB ' lik bir varsayılan değer içerir.

Bu özniteliğin değerini saptamak için, kuyruk yöneticisi nesnesindeki MQINQ çağrısını kullanın. Büyük iletiler için bu değeri değiştirebilirsiniz.

Kuyruğun *MaxMsgLength* özniteliği, kuyruğa koyabileceğiniz ileti boyutu üst sınırını belirler. Bu özniteliğin değerinden büyük bir boyuta sahip bir ileti yerleştirmeye çalışırsanız, MQPUT çağrılarınız başarısız olur. Bir iletiyi uzak bir kuyruğa yerleştiriyorsanız, başarıyla yerleştirebileceğiniz ileti boyutu üst sınırı, uzak kuyruğun *MaxMsgLength* özniteliği tarafından belirlenir ve bu ileti, iletinin hedef rotasına ve kullanılan kanallara ilişkin ara iletim kuyruklarının herhangi bir ara iletim kuyruğundan saptanır.

Bir MQPUT işlemi için, iletinin büyüklüğü hem kuyruğun, hem de kuyruk yöneticisinin *MaxMsgLength* özniteisinden küçük ya da ona eşit olmalıdır. Bu özniteliklerin değerleri bağımsızdır, ancak kuyruğun *MaxMsgLength* ' unu kuyruk yöneticisinden küçük ya da ona eşit bir değere ayarlamanız önerilir.

WebSphere MQ , aşağıdaki durumlarda iletilere üstbilgi bilgileri ekler:

- Uzak bir kuyruğa ileti koyduğunuzda, WebSphere MQ iletiye bir iletim üstbilgisi yapısı (MQXQH) ekler. Bu yapı, hedef kuyruğun adını ve sahip olan kuyruk yöneticisini içerir.
- WebSphere MQ bir iletiyi uzak bir kuyruğa teslim edemiyorsa, iletiyi ölü harf (teslim edilemeyen ileti) kuyruğuna yerleştirmeyi dener. İletiye MQDLH yapısı ekler. Bu yapı, hedef kuyruğun adını ve iletinin ölü-mektup kuyruğuna konmasının nedenini içerir.
- Birden çok hedef kuyruğa ileti göndermek istiyorsanız, WebSphere MQ iletiye bir MQDH üstbilgisi ekler. Bu, bir iletim kuyruğunda yer alan bir dağıtım listesine ait olan bir iletide bulunan verileri açıklar. İleti uzunluğu üst sınırı için en iyi değer seçilirken bunu dikkate alın.
- İleti bir bölümse ya da bir gruptaki bir iletiyse, WebSphere MQ bir MQMDE ekleyebilir.

Bu yapılar MQDH ve MQMDEiçinde açıklanmıştır.

İletilerinizde bu kuyruklar için izin verilen büyüklük üst sınırı varsa, bu üstbilgilerin eklenmesi, iletilerin artık çok büyük olduğu için, koyma işlemlerinin başarısız olduğu anlamına gelir. Koyma işlemlerinin başarısız olma olasılığını azaltmak için:

- İletilerinizin boyutunu, iletim ve ölü harf kuyruklarının *MaxMsgLength* öznitelikten daha küçük bir hale getiriniz. En azından MQ_MSG_HEADER_LENGTH değişmezinin (büyük dağıtım listeleri için daha fazla) değerine izin verin.
- Ölü-harfli kuyruğun *MaxMsgLength* özniteliğinin, ölü harf kuyruğunun sahibi olan kuyruk yöneticisinin *MaxMsgLength* ile aynı olarak ayarlandığından emin olun.

Kuyruk yöneticisine ilişkin öznitelikler ve kuyruğa alma değişmezleri, kuyruk yöneticisine ilişkin özniteliklerialtında açıklanmıştır.

İleti koyma: İleti tanıtıcılarını kullanma

MQPMO yapısında iki ileti tanıtıcısı bulunur, *OriginalMsgHandle* ve *NewMsgHandle*(Yeni Msg) tanıtıcısı. Bu ileti tanıtıcıları arasındaki ilişki, MQPMO *Action* alanının değeriyle tanımlanır.

Tüm ayrıntılar için bkz. Eylem (MQUZE). Bir iletiyi koymak için bir ileti tanıtıcısı gerekli değildir. Bunun amacı, özellikleri bir iletiyle ilişkilendirmeniz, bu nedenle yalnızca ileti özelliklerini kullanıyorsanız gereklidir.

İletileri Uzak Kuyruğa Koyma

Bir iletiyi uzak bir kuyruğa (örneğin, uygulamanızın bağlı olduğu bir kuyruk yöneticisine ait bir kuyruk) yerel bir kuyruk yerine koymak istediğinizde, daha fazla dikkat edilmesi gereken tek dikkat, kuyruğun adını açtığınızda adın nasıl belirtildiğini belirtir.Bu, “Uzak Kuyrukların Açılması” sayfa 214içinde açıklanmaktadır. Yerel bir kuyruk için MQPUT ya da MQPUT1 çağrısını kullanma şekliniz bir değişiklik yok.

Uzak ve iletim kuyruklarının kullanılmasına ilişkin ek bilgi için WebSphere MQ distributed-messaging tekniklerini başlıklı konuya bakın.

İletiye ilişkin özelliklerin ayarlanması

Ayarlamak istediğiniz her özellik için MQSETMP ' yi çağırın. İletiyi yerleştirdiğinizde, MQPMO yapısının ileti tanıtıcısı ve işlem alanları ayarlanır.

Özellikleri bir iletiyle ilişkilendirmek için, iletinin bir ileti tanıtıcısı olması gerekir. MQCRTMH işlev çağrısını kullanarak bir ileti tanıtıcısı yaratın. Ayarlamak istediğiniz her özellik için bu ileti tanıtıcısını belirterek MQSETMP ' yi çağırın. MQSETMP kullanımını göstermek için bir örnek program (amqsstma.c) sağlanmıştır.

Bu yeni bir iletiyse, bu iletiyi bir kuyruğa koyduğunuzda, MQPUT ya da MQPUT1kullanarak, MQPMO ' daki OriginalMsgHandle alanını bu ileti tanıtıcısı değerine ayarlayın ve MQPMO Action alanını MQACTP_NEW olarak ayarlayın (varsayılan değer budur).

Bu, önceden aldığınız bir iletiyse ve şimdi bu iletiyi iletiyor ya da yanıtladınız ya da buna yanıt olarak bir rapor gönderiyorsanız, özgün ileti tanıtıcısını MQPMO ' nun OriginalMsgHandle alanına ve NewMsgHandle alanındaki yeni ileti tanıtıcısını yerleştirdiniz. İşlem alanını, uygun olduğu şekilde MQACTP_FORWARD, MQACTP_REPLY ya da MQACP_REPORT olarak ayarlayın.

Bir MQRFH2 üstbilgisinde önceden aldığınız bir iletiden özellikler varsa, bu iletiyi MQBUFMH çağrısını kullanarak ileti işleyici özelliklerine dönüştürebilirsiniz.

İletinizi, ileti özelliklerini işleyemeyen WebSphere MQ Sürüm 7.0sürümünden önceki bir düzeydeki kuyruk yöneticisinde bir kuyruğa yerleştiriyorsanız, özelliklerin nasıl işleneceğini belirtmek için kanal tanımındaki PropertyControl parametresini ayarlayabilirsiniz.

Bağlam bilgilerini denetleme

Bir iletiyi kuyruğa koymak için MQPUT ya da MQPUT1 çağrısını kullandığınızda, kuyruk yöneticisinin ileti tanımlayıcısına varsayılan bağlam bilgileri eklemesini belirleyebilirsiniz. Uygun yetki düzeyine sahip uygulamalar ek bağlam bilgileri ekleyebilir. Bağlam bilgilerini denetlemek için, MQPMO yapısındaki seçenekler alanını kullanabilirsiniz.

Bağlam bilgilerini denetlemek için, MQPMO yapısındaki *Options* alanını kullanın.

Bunu yapmazsanız, kuyruk yöneticisi, ileti tanımlayıcısında, iletiniz için oluşturduğu tanıtıcı ve bağlam bilgileriyle önceden olabilecek bağlam bilgilerini yazar. Bu, MQPMO_DEFAULT_CONTEXT seçeneğinin belirlendiği şekliyle aynıdır.Yeni bir ileti yarattığınızda (örneğin, bir sorgu ekranından kullanıcı girişi işlenirken) bu varsayılan bağlam bilgilerinin olmasını isteyebilirsiniz.

İletinizle ilişkilendirilmiş bir bağlam bilgisi istemiyorsanız, MQPMO_NO_CONTEXT seçeneğini kullanın. Bağlam olmadan bir ileti yerleştirilirken, IBM WebSphere MQ tarafından yapılan tüm yetki denetimleri boş bir kullanıcı kimliği kullanılarak yapılır. Boş bir kullanıcı kimliği, IBM WebSphere MQ kaynaklarına belirtik yetki atanamaz; ancak, 'kimse' grubunun bir üyesi olarak işlem görür. nobodyözel grubuna ilişkin ayrıntılar için Kurulabilir hizmetler arabirimi başvuru bilgileribaşlıklı konuya bakın.

İletinizle ilişkilendirilmiş bir bağlam bilgisi istemiyorsanız, MQPMO_NO_CONTEXT seçeneğini kullanın.

Bu konunun aşağıdaki bölümleri, kimlik bağlamı, kullanıcı bağlamı ve tüm bağlamın kullanımını açıklar.

- “Kimlik bağlamı geçirme” sayfa 222
- “Kullanıcı bağlamı geçirme” sayfa 223
- “Tüm bağlamın geçirilmesi” sayfa 223
- “Kimlik bağlamının ayarlanması” sayfa 223
- “Kullanıcı bağlamını ayarlama” sayfa 223
- “Tüm bağlamın ayarlanması” sayfa 223

Kimlik bağlamı geçirme

Genel olarak, programlar, bir uygulamanın son hedefine ulaşıncaya kadar, bir uygulama çevresinde kimlik bağlamı bilgilerini iletiden iletiye iletmelidir.

Programlar, verileri her değiştirişlerinde kaynak bağlamı bilgisini değiştirmelidir. Ancak, herhangi bir bağlam bilgisini değiştirmek ya da ayarlamak isteyen uygulamaların uygun yetki düzeyine sahip olması gerekir. Kuyruk yöneticisi, uygulamalar kuyrukları alarken bu yetkiyi denetler; MQOPEN çağrısına ilişkin uygun bağlam seçeneklerini kullanma yetkisi olmalıdır.

Uygulamanız bir ileti alırsa, iletiden verileri işler, daha sonra değiştirilen verileri başka bir iletiye koyar (olasılıkla başka bir uygulama tarafından işlenebilir), uygulamanın kimlik bağlamı bilgilerini özgün iletiden yeni iletiye geçirmesi gerekir. Kuyruk yöneticisinin kaynak bağlamı bilgilerini yaratmasına izin verebilirsiniz.

Bağlam bilgilerini özgün iletiden kaydetmek için, iletiyi almak için kuyruğu açtığınızda MQOO_SAVE_ALL_CONTEXT seçeneğini kullanın. Bu, MQOPEN çağrısıyla birlikte kullandığınız diğer seçeneklerin yanı sıra. Ancak, yalnızca iletiye göz attığınızda bağlam bilgilerini kaydedemezsiniz.

İkinci iletiyi oluşturduğunuzda:

- MQOO_PASS_IDENTITY_CONTEXT seçeneğini kullanarak kuyruğu açın (MQOO_OUTPUT seçeneğinin yanı sıra).
- İletiye ilişkin seçenekler yapısının *Context* alanında, bağlam bilgilerini sakladığınız kuyruğun tanıtıcısını verin.
- Put-message options yapısının *Options* alanında, MQPMO_PASS_IDENTITY_CONTEXT seçeneğini belirtin.

Kullanıcı bağlamı geçirme

Yalnızca kullanıcı bağlamını geçmeyi seçemezsiniz. Bir ileti yerleştirilirken kullanıcı bağlamını geçirmek için, MQPMO_PASS_ALL_CONTEXT belirtin. Kullanıcı bağlamındaki özellikler, kaynak bağlamla aynı şekilde geçirilir.

Bir MQPUT ya da MQPUT1 gerçekleştiğinde ve bağlam geçiriliyorsa, kullanıcı bağlamındaki tüm özellikler, alınan iletiden alınan iletiye iletilir. Koyma uygulamasının değiştirilmiş olduğu kullanıcı bağlamı özellikleri özgün değerleriyle konmuştur. Koyma uygulamasının silmiş olduğu kullanıcı bağlamı özellikleri, koyma iletisinde geri yüklenir. Ekleme uygulamasının iletiye eklediği kullanıcı bağlamı özellikleri korunur.

Tüm bağlamın geçirilmesi

Uygulamanız bir ileti alıyorsa ve ileti verilerini (değiştirilmemiş) başka bir iletiye koyarsa, uygulamanın tümünü (kimlik, kaynak ve kullanıcı) bağlam bilgilerini özgün iletiden yeni iletiye iletmesi gerekir. Bunu yapabilen bir uygulama örneği, iletileri bir kuyruktan diğerine taşıyan bir ileti modemdir.

MQOO_PASS_PASS_ALL_CONTEXT ve MQPMO_PASS_ALL_CONTEXT koyma iletisi seçeneğini kullanmanız dışında, kimlik bağlamını geçirmek için aynı yordamı izleyin.

Kimlik bağlamının ayarlanması

Bir ileti için kimlik bağlamı bilgilerini ayarlamak istiyorsanız:

- MQOO_SET_IDENTITY_CONTEXT seçeneğini kullanarak kuyruğu açın.
- MQPMO_SET_IDENTITY_CONTEXT seçeneğini belirterek, iletiyi kuyruğa koyun. İleti tanımlayıcısında, gereksinim duyduğunuz kimlik bağlamı bilgilerini belirtin.

Not: MQOO_SET_IDENTITY_CONTEXT ve MQPMO_SET_IDENTITY_CONTEXT seçeneklerini kullanarak, kimlik bağlamı alanlarının bazılarını (ancak tümü değil) ayarladığınızda, kuyruk yöneticisinin diğer alanların hiçbirini ayarlamadığını fark etmek önemlidir.

İleti bağlamı seçeneklerinden herhangi birini değiştirmek için, aramayı yayınlamak için gereken yetkilerin olması gerekir. Örneğin, MQOO_SET_IDENTITY_CONTEXT ya da MQPMO_SET_IDENTITY_CONTEXT kullanabilmek için `+setid` iznine sahip olmanız gerekir.

Kullanıcı bağlamını ayarlama

Kullanıcı bağlamında bir özellik ayarlamak için, MQSETMP çağrısını gerçekleştirdiğinizde, ileti özelliği tanımlayıcısının (MQPD) Bağlam alanını MQPD_USER_CONTEXT olarak ayarlayın.

Kullanıcı bağlamında bir özellik ayarlamak için özel bir yetkiye sahip olmamanız gerekir. Kullanıcı bağlamında MQOO_SET_ * ya da MQPMO_SET_ * bağlam seçenekleri yok.

Tüm bağlamın ayarlanması

Bir ileti için hem kimlik, hem de kaynak bağlamı bilgilerini ayarlamak istiyorsanız:

1. MQOO_SET_ALL_CONTEXT seçeneğini kullanarak kuyruğu açın.
2. MQPMO_SET_ALL_CONTEXT seçeneğini belirterek, iletiyi kuyruğa koyun. İleti tanımlayıcısında, gereksinim duyduğunuz kimlik ve kaynak bağlamı bilgilerini belirtin.

Her bir bağlam ayarı tipi için uygun yetki gereklidir.

İlgili kavramlar

“İleti bağlamı” sayfa 37

İleti bağlamı bilgileri, iletiyi alan uygulamanın, iletiyi oluşturan iletiyi bulmasını sağlayan bir uygulamaya olanak tanır.

İlgili başvurular

“İleti bağlamına ilişkin MQAç seçenekleri” sayfa 213

Bir iletiyi bir kuyruğa koyduğunuzda bağlam bilgilerini bir iletiyle ilişkilendirebilmek istiyorsanız, kuyruğu açtığınızda ileti bağlamı seçeneklerinden birini kullanmanız gerekir.

MQPUT1 çağrısını kullanarak bir ileti kuyruğa konması

Kuyruğa tek bir ileti koyduktan hemen sonra kuyruğu kapatmak istediğinizde MQPUT1 çağrısını kullanın. Örneğin, bir sunucu uygulaması, farklı kuyrukların her birine yanıt gönderirken MQPUT1 çağrısını kullanacaktır.

MQPUT1 , MQPUT ve onu izleyen MQCLOSE ' yi çağırarak işlevsel olarak eşdeğerdir. MQPUT ve MQPUT1 çağrılarına ilişkin sözdizimindeki tek fark, MQPUT için bir nesne tanıtıcısı belirtmenizi sağlar; MQPUT1 için, MQPAN ' da tanımlandığı gibi bir nesne tanımlayıcı yapısı (MQOD) belirtiyorsunuz (bkz. “Nesnelerin tanımlanması (MQOD yapısı)” sayfa 208). Bunun nedeni, MQPUT1 'in açılması gerektiği kuyruğunuz hakkında bilgi vermeniz gerektiğinden, MQPUT' u çağırdığınızda, kuyruk zaten açık olmalıdır.

MQPUT1 çağrısına giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı.
- Açmak istediğiniz nesneye ilişkin bir açıklama. Bu, bir nesne tanımlayıcı yapısı biçimidir (MQOD).
- Kuyruğa koymak istediğiniz iletinin açıklaması. Bu, ileti tanımlayıcı yapısı (MQMD) biçimidir.
- Put-message options structure (MQPMO) yapısı biçiminde bilgileri denetleyin.
- İleti içinde yer alan verilerin uzunluğu (MQlong).
- İleti verilerinin adresi.

The output from MQPUT1 is:

- Tamamlanma kodu
- Neden kodu

Arama başarıyla tamamlanırsa, seçenek yapılarınızı ve ileti tanımlayıcı yapınızı da döndürür. Çağrı, seçenek yapınızı, iletinin adını ve iletinin gönderildiği kuyruk yöneticisini gösterecek şekilde değiştirir. Kuyruk yöneticisinin koymakta olduğunuz iletinin tanıtıcısı için benzersiz bir değer oluşturmasını isteseniz (MQMD yapısının *MsgId* alanında ikili sıfır belirleyerek), arama bu yapıyı size döndürmeden önce *MsgId* alanına değeri ekler.

Not: Model kuyruğu adıyla MQPUT1 kullanamazsınız; ancak, bir model kuyruğu açıldıktan sonra, dinamik kuyruk için bir MQPUT1 komutu verebilirsiniz.

MQPUT1 için altı giriş parametresi şunlardır:

Hconn
Bu bir bağlantı tanıtıcısı. CICS uygulamaları için, MQHC_DEF_HCONN değişmezini (sıfır değerine sahip) belirtebilir ya da MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanabilirsiniz. Diğer programlar için, her zaman MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanın.

ObjDesc
Bu bir nesne tanımlayıcısı yapısıdır (MQOD).

ObjectName ve *ObjectQMgrName* alanlarında, ileti koymak istediğiniz kuyruğun adını ve bu kuyruğun sahibi olan kuyruk yöneticisinin adını verin.

Model kuyruklarını kullanamadığı için, MQPUT1 çağrısı için *DynamicQName* alanı yoksayılır.

Kuyruğu açmak için yetki vermek üzere kullanılacak diğer bir kullanıcı kimliğini aday göstermek istiyorsanız, *AlternateUserId* alanını kullanın.

MsgDesc
Bu bir ileti tanımlayıcı yapısıdır (MQMD). MQPUT çağrısıyla olduğu gibi, kuyruğa koymakta olduğunuz iletiyi tanımlamak için bu yapıyı kullanın.

PutMsgOpts
Bu bir put-message options yapısıdır (MQPMO). Bunu, MQPUT çağrısı için kullanırken kullanın (bkz. “MQPMO yapısını kullanarak seçenekleri belirtme” sayfa 217).

Options alanı sıfır olarak ayarlandığında, kuyruk yöneticisi, kuyruğa erişim yetkisi için sınamalar gerçekleştirirken kendi kullanıcı kimliğinizi kullanır. Ayrıca, kuyruk yöneticisi, MQOD yapısının *AlternateUserId* alanında verilen diğer kullanıcı kimliğini yoksayar.

BufferLength
Bu, iletinizin uzunluğidir.

Buffer
Bu, iletinizin metnini içeren arabellek alanıdır.

Kümeleri kullandığınızda, MQPUT1 , MQOO_BIND_NOT_FIXED etkin olduğu gibi çalışır. İletinin nereye gönderileceğini belirlemek için, uygulamaların MQPMO yapısındaki çözülmüş alanları MQPO yapısında kullanması gerekir. Ek bilgi için Kuyruk yöneticisi kümesinin yapılandırılması başlıklı konuya bakın.

MQPUT1içinde MQPUT1 çağrısının bir açıklaması vardır.

Dağıtım listeleri

z/OSiçin WebSphere MQ ' da desteklenmez. Dağıtım listeleri, tek bir MQPUT ya da MQPUT1 çağrısında birden çok hedefe bir ileti koymanızı sağlar.Tek bir MQOPER çağrısı birden çok kuyruk açabilir ve sonra tek bir MQPUT çağrısı bu kuyrukların her birine bir ileti yerleştirebilir. Bu süreç için kullanılan MQI yapılarından alınan bazı genel bilgiler, dağıtım listesinde yer alan tek tek hedeflerle ilgili belirli bilgiler tarafından geçersiz kılınabilir.

V 7.5.0.8

Uyarı: Dağıtım listeleri, konu nesnelerini gösteren diğer ad kuyruklarını kullanmaz. Version 7.5.0, Fix Pack 8' tan, bir diğer ad kuyruğu bir dağıtım listesindeki bir konu nesnesini gösteriyorsa, IBM WebSphere MQ , MQRC_ALIAS_BASE_Q_TYPE_ERROR değerini döndürür.

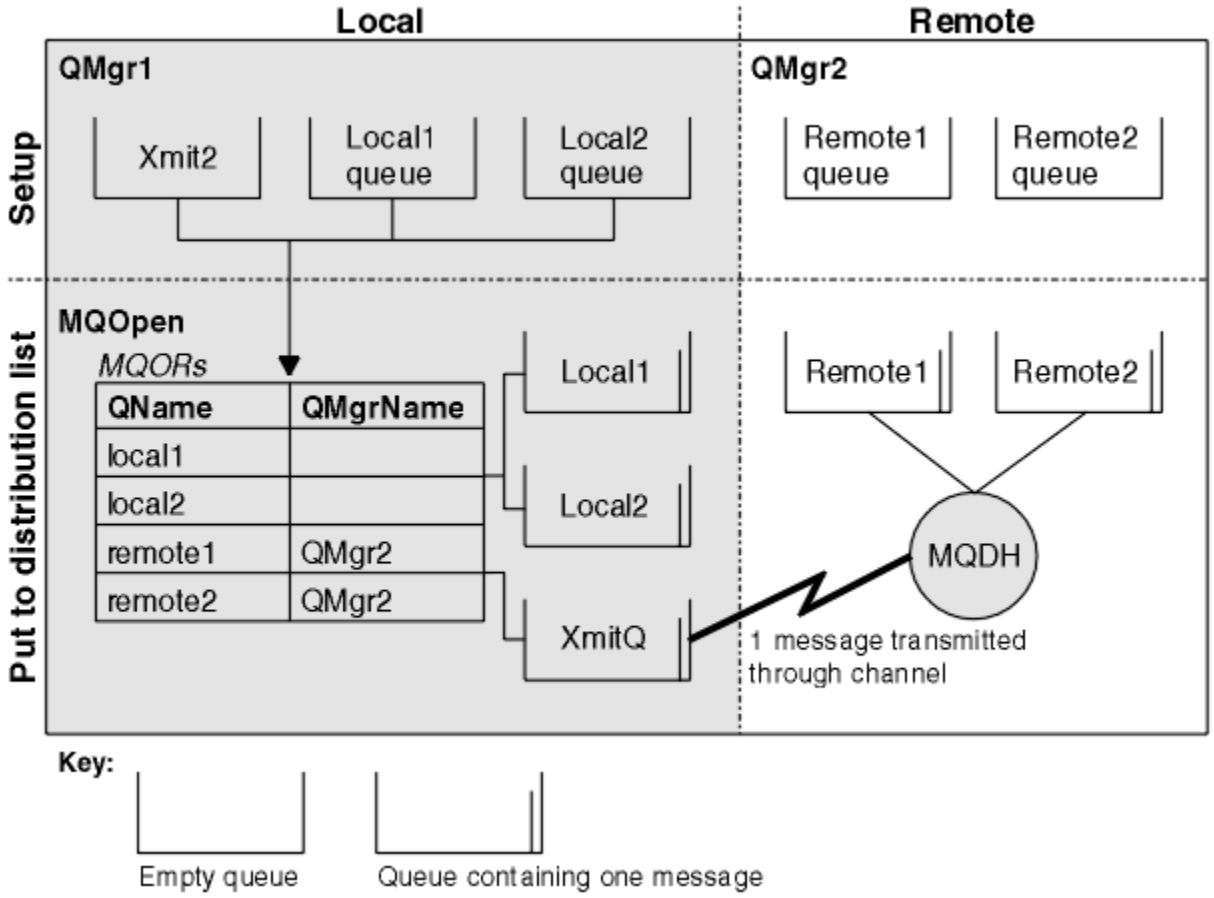
Bir MQOPED çağrısı yayınlandığında, Nesne Tanımlayıcısından (MQOD) soysal bilgiler alınır. *Version* alanında MQOD_VERSION_2 değerini ve *RecsPresent* alanında sıfırdan büyük bir değer belirtirseniz, *Hobj* , kuyruk yerine bir listenin (bir ya da daha çok kuyruğun) tanıtıcı değeri olarak tanımlanabilir. Bu durumda, hedef ayrıntılarını (yani, *ObjectName* ve *ObjectQMgrName*) veren nesne kayıtları (MQORS) aracılığıyla belirli bilgiler verilir.

Nesne tanıtıcısı (*Hobj*) MQPUT çağrısına iletilir ve tek bir kuyruk yerine bir listeye girmenize olanak tanır.

Kuyruklara bir ileti konduğunda (MQPUT) soysal bilgiler, Put Message Option structure (MQPMO) ve Message Descriptor (MQMD)) olanağından alınır. Belirli bilgiler, Put Message Records (MQPMR ' ler) biçiminde verilir.

Yanıt Kayıtları (MQRR), her hedef kuyruğa özgü bir tamamlanma kodu ve neden kodu alabilir.

Şekil 29 sayfa 226 , dağıtım listelerinin nasıl çalıştığını gösterir.



Şekil 29. Dağıtım listeleri nasıl çalışır

Dağıtım listelerini açma

Bir dağıtım listesi açmak için MQOPEN çağrısını kullanın ve arama seçeneklerini kullanarak, listeyle ne yapmak istediğinizi belirtin.

MQOPER ' a giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı (açıklama için “İletileri Kuyruğa Koyma” sayfa 215 konusuna bakın)
- Nesne Tanımlayıcı yapısındaki soysal bilgiler (MQOD)
- Nesne Kaydı yapısını (MQOR) kullanarak açmak istediğiniz her kuyruğun adı.

MQXX_ENCODE_CASE_ONE open komutunun çıkışı şöyledir:

- Dağıtım listesine erişiminizi temsil eden bir nesne tanıtıcısı
- Soysal bir tamamlanma kodu
- Soysal neden kodu
- Yanıt Kayıtları (isteğe bağlı), her hedef için bir tamamlanma kodu ve neden içeren

MQOD yapısının kullanılması

Açmak istediğiniz kuyrukları tanımlamak için MQOD yapısını kullanın.

Bir dağıtım listesi tanımlamak için, *Version* alanında MQOD_VERSION_2 belirtmeli, *RecsPresent* alanında sıfırdan büyük bir değer ve *ObjectType* alanında MQOT_Q belirtilmelidir. MQOD yapısının tüm alanlarının bir açıklaması için bkz. MQOD .

MQOR yapısının kullanılması

Her hedef için bir MQOR yapısı sağlayın.

Yapı, hedef kuyruğu ve kuyruk yöneticisi adlarını içerir. MQOD ' daki *ObjectName* ve *ObjectQMgrName* alanları dağıtım listeleri için kullanılmaz. Bir ya da daha çok nesne kaydı olmalıdır. *ObjectQMgrName* boş bırakılırsa, yerel kuyruk yöneticisi kullanılır. Bu alanlarla ilgili ek bilgi için bkz. ObjectName ve ObjectQMgrAd .

Hedef kuyrukları iki şekilde belirtebilirsiniz:

- *ObjectRecOffset*göreli konum alanını kullanarak.

  In this case, the application must declare its own structure containing an MQOD structure, followed by the array of MQOR records (with as many array elements as are needed), and set *ObjectRecOffset* to the offset of the first element in the array from the start of the MQOD. Bu göreli konumun doğru olduğundan emin olun.

  Bu programlar, uygulamanın çalıştığı tüm ortamlarda kullanılabiliyorsa, programlama dili tarafından sağlanan yerleşik tesislerin kullanılması önerilir. Aşağıdaki kod, COBOL programlama diline ilişkin bu tekniği göstermektedir:

```
01  MY-OPEN-DATA.
    02 MY-MQOD.
       COPY CMQODV.
    02 MY-MQOR-TABLE OCCURS 100 TIMES.
       COPY CMQORV.
    MOVE LENGTH OF MY-MQOD TO MQOD-OBJECTRECOFFSET.
```

  Diğer bir seçenek olarak, programlama dili, ilgili tüm ortamlardaki gerekli yerleşik olanakları desteklemiyorsa, MQOD_CURRENT_LENGTH değişmezini kullanın. Aşağıdaki kod bu tekniği gösterir:

```
01  MY-MQ-CONSTANTS.
    COPY CMQV.
01  MY-OPEN-DATA.
    02 MY-MQOD.
       COPY CMQODV.
    02 MY-MQOR-TABLE OCCURS 100 TIMES.
       COPY CMQORV.
    MOVE MQOD-CURRENT-LENGTH TO MQOD-OBJECTRECOFFSET.
```

  Ancak, bu işlev yalnızca MQOD yapısı ve MQOR kayıtları dizisi bitişik olduğunda doğru çalışır; derleyici, MQOD ile MQOR dizisi arasına atlamalı byte ekler eklediyse, bunlar *ObjectRecOffset*içinde saklanan değere eklenmelidir.

  İşaretçi veri tipini desteklemeyen programlama dilleri için *ObjectRecOffset* kullanılması önerilir ya da gösterge verileri tipi, farklı ortamlara (örneğin, COBOL programlama dili) portatif olmayan bir biçimde uygulanır.

- *ObjectRecPtr*işaretçi alanını kullanarak.

  Bu durumda uygulama, MQOR yapılarının dizisini, MQOD yapısından ayrı olarak bildirebilir ve *ObjectRecPtr* ' yi dizinin adresine ayarlayabilir. Aşağıdaki kod C programlama dili için bu tekniği göstermektedir:

```
MQOD MyMqod;
MQOR MyMqor[100];
MyMqod.ObjectRecPtr = MyMqor;
```

  İşaretçi veri tipini farklı ortamlara (örneğin, C programlama dili gibi) destekleyen programlama dilleri için *ObjectRecPtr* ' nin kullanılması önerilir.

Seçtiğiniz teknik, *ObjectRecOffset* ve *ObjectRecPtr*' den birini kullanmanız gerekir; her ikisi de sıfırsa ya da her ikisi de sıfırsa, çağrı neden kodu MQRC_OBJECT_RECORDS_ERROR ile başarısız olur.

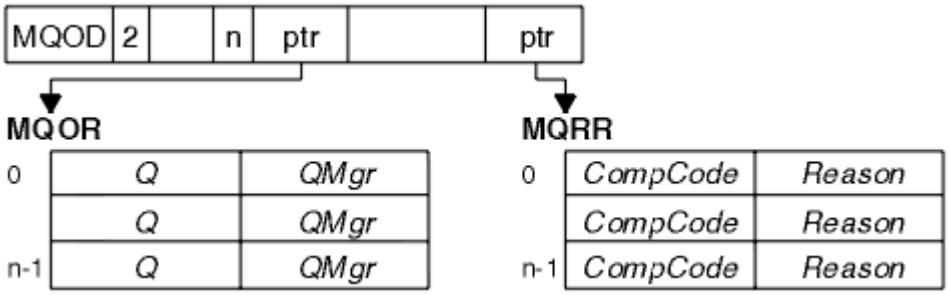
MQRR yapısının kullanılması

Bu yapılar hedef belirtimidir; her Yanıt Kaydı, dağıtım listesinin her kuyruğu için bir *CompCode* ve *Reason* alanı içerir. Herhangi bir sorunun yalanı ayırt edebilmenizi sağlamak için bu yapıyı kullanmanız gerekir.

Örneğin, MQRC_MULTIPLE_REASONS bir neden kodu alırsanız ve dağıtım listeniz beş hedef kuyruk içeriyorsa, bu yapıyı kullanmamanız durumunda sorunların hangi kuyruklara uygulacağını bilmeyeceksiniz. Ancak, her hedef için bir tamamlama kodunuz ve neden kodunuz varsa, hataları daha kolay bulabilirsiniz.

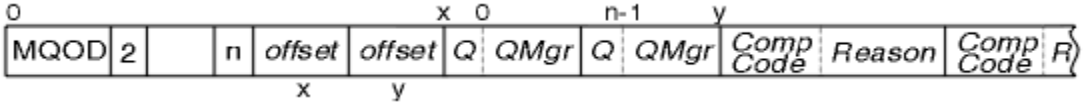
MQRR yapısıyla ilgili ek bilgi için MQRR konusuna bakın.

Şekil 30 sayfa 228 içinde, C içinde bir dağıtım listesinin nasıl açılabileceğinin gösterilmektedir.



Şekil 30. C içinde bir dağıtım listesinin açılması

Şekil 31 sayfa 228 , COBOL ' da bir dağıtım listesinin nasıl açılabileceğini gösterir.



Şekil 31. COBOL ' da bir dağıtım listesinin açılması

MQOPER seçeneklerinin kullanılması

Bir dağıtım listesi açarken aşağıdaki seçenekleri belirleyebilirsiniz:

- MQOO_OUTPUT
- MQOO_FAIL_IF_QUIESCING (isteğe bağlı)
- MQOO_ALTERNATE_USER_AUTHORITY (isteğe bağlı)
- MQOO_ * _CONTEXT (isteğe bağlı)

Bu seçeneklerin açıklaması için bkz. “Nesnelerin açılması ve kapatılması” sayfa 206 .

İletileri Dağıtım Listesine Koyma

Bir dağıtım listesine ileti koymak için, MQPUT ya da MQPUT1' i kullanabilirsiniz.

Giriş olarak şu bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı (açıklama için “İletileri Kuyruğa Koyma” sayfa 215 konusuna bakın).
- Bir nesne tanıtıcısı. Bir dağıtım listesi MQOPEN kullanılarak açılırsa, *Hobj* yalnızca listeye koymanıza izin verir.
- Bir ileti tanımlayıcı yapısı (MQMD). Bu yapının bir açıklaması için bkz. MQMD .
- Put-message option structure (MQPMO) biçiminde bilgileri denetleyin. MQPMO yapısının alanlarının tamamlanmasına ilişkin bilgi için bkz. “MQPMO yapısını kullanarak seçenekleri belirtme” sayfa 217 .
- İleti Koyma Kayıtları (MQPMR) biçimindeki denetim bilgileri.
- İleti içinde yer alan verilerin uzunluğu (MQlong).
- İleti verilerinin kendisi.

Çıkış:

- Tamamlanma kodu
- Neden kodu
- Yanıt Kayıtları (isteğe bağlı)

MQPMR yapısının kullanılması

Bu yapı isteğe bağlıdır ve MQMD ' de önceden tanımlananlardan farklı bir şekilde tanımlamak isteyebileceğiniz bazı alanlar için hedefe özgü bilgileri verir.

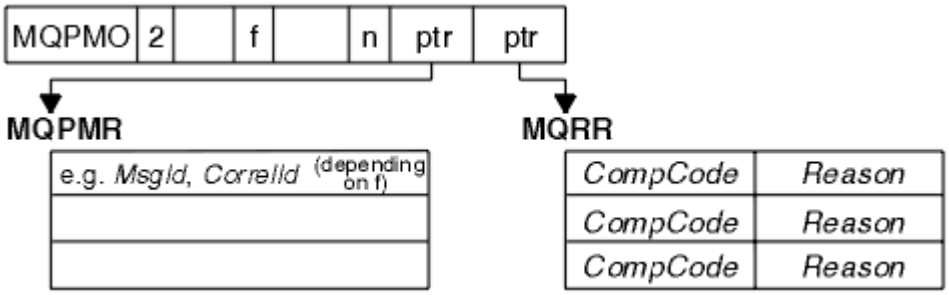
Bu alanlara ilişkin açıklamalar için MQPMRbaşlıklı konuya bakın.

Her kaydın içeriği, MQPMO ' nun *PutMsgRecFields* alanında verilen bilgilere bağlıdır. Örneğin, AMQSPTL0.C (açıklama için bkz. “Dağıtım Listesi örnek programı” sayfa 122) dağıtım listelerinin kullanımını gösteren örnek, MQPMR ' de *MsgId* ve *CorrelId* için değer sağlamayı seçer. Örnek programın bu bölümü şu şekilde görünür:

```
typedef struct
{
MQBYTE24 MsgId;
MQBYTE24 CorrelId;
} PutMsgRec;
...
/**********************
MQLONG PutMsgRecFields=MQPMRF_MSG_ID | MQPMRF_CORREL_ID;
```

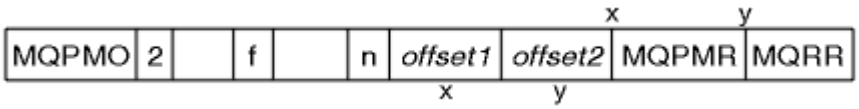
Bu, dağıtım listesinin her hedefi için *MsgId* ve *CorrelId* ' in sağlandığına işaret eder. Put Message Records, bir dizi olarak sağlanır.

Şekil 32 sayfa 229 , C içindeki bir dağıtım listesine nasıl bir ileti yerleştirebileceğinin gösterir.



Şekil 32. C içindeki bir dağıtım listesine ileti konması

Şekil 33 sayfa 229 , COBOL ' da bir dağıtım listesine nasıl bir ileti yerleştirebileceğinin gösterir.



Şekil 33. COBOL ' de bir dağıtım listesine ileti konması

MQPUT1komutunu kullanma

MQPUT1kullanıyorsanız, aşağıdaki noktaları göz önünde bulundurun:

1. *ResponseRecOffset* ve *ResponseRecPtr* alanlarının değerleri boş değerli ya da sıfır olmalıdır.
2. Gerekirse, Yanıt Kayıtları MQOD ' den adreslenmelidir.

Put çağrılarının başarısız olduğu bazı durumlar

Bir MQOPEN ve bir MQPUT çağrısıyla arasındaki aralık sırasında bir komutun FORCE seçeneği kullanılarak bir kuyruğun bazı öznitelikleri değiştirilirse, MQPUT çağrısı başarısız olur ve MQRC_OBJECT_CHANGED neden kodunu döndürür.

Kuyruk yöneticisi, nesne tanıtıcısını artık geçerli değil olarak işaretler. Bu durum, bir MQPUT1 çağrısı işlenirken ya da değişikliklerin kuyruk adının çözümlediği herhangi bir kuyruk üzerinde uygulanırsa, bu durum da oluşur. Bu şekilde, tanıtıcıyı etkileyen öznitelikler, MQOPEN' da MQOPER çağrısının tanımında listelenir. Aramanız MQRC_OBJECT_CHANGED neden kodunu döndürürse, kuyruğu kapatın, yeniden açın ve sonra bir ileti yeniden yerleştirmeyi deneyin.

İletileri (ya da kuyruk adının çözümlendiği herhangi bir kuyruğu) yerleştirmeye çalıştığınız bir kuyruk için bir işlem engellenirse, MQPUT ya da MQPUT1 çağrısı başarısız olur ve MQRC_PUT_INHIBITED neden kodunu döndürür. Aramayı daha sonra denerseniz, başka programlar kuyrukların özniteliklerini düzenli olarak değiştiriyorsa, iletiyi daha sonra başarılı bir şekilde gerçekleştirebilseniz de, iletiyi başarıyla yerleştirebilirsiniz.

Furthemore, iletinizi yerleştirmeye çalıştığınız kuyruk dolu olursa, MQPUT ya da MQPUT1 çağrısı başarısız olur ve MQRC_Q_FULL değerini döndürür.

Bir dinamik kuyruk (geçici ya da kalıcı) silindiyse, MQPUT çağrıları önceden edinilmiş bir nesne tanıtıcısı kullanılarak başarısız olur ve MQRC_Q_DELETED neden kodunu döndürür. Bu durumda, nesne tanıtıcısını kapatmanız artık size herhangi bir faydası olmadığı için iyi bir uygulamadır.

Dağıtım listeleri durumunda, tek bir istekte birden çok tamamlanma kodu ve neden kodu ortaya çıkabilir. Bu işlem, yalnızca MQOUT ve MQPUT üzerindeki *CompCode* ve *Reason* çıkış alanları kullanılarak işlenemez.

Birden çok hedefe ileti koymak için dağıtım listeleri kullandığınızda, Yanıt Kayıtları her hedef için belirli *CompCode* ve *Reason* ' yi içerir. MQCC_FAILED tamamlanma kodu alırsanız, herhangi bir hedef kuyruğa başarıyla herhangi bir ileti konmaz. Tamamlanma kodu MQCC_UYARI ise, ileti bir ya da daha çok hedef kuyruktan başarıyla konabiliyor. Dönüş kodu MQRC_MULTIPLE_REASONS değerini alırsanız, neden kodları her hedef için aynı değildir. Bu nedenle, bir hataya neden olan kuyruğu ya da kuyrukları saptamanız için MQRR yapısının kullanılması önerilir; böylece, bir hataya neden olan kuyruklar ve nedenler de vardır.

Kuyruktan Ileti Alınması

Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

Bir kuyruktan iki şekilde ileti alabilirsiniz:

1. Bir iletiyi diğer programların artık görebilmesi için kuyruktan kaldırabilirsiniz.
2. Bir iletiyi kopyalayabilir ve özgün iletiyi kuyruğun üzerine bırakabilirsiniz. Bu, *göz atma*olarak bilinir. Bu iletiyi göz attığınızda, iletiyi kaldırabilirsiniz.

Her iki durumda da, MQGET çağrısını kullanıyorsunuz, ancak önce uygulamanızın kuyruk yöneticisine bağlı olması ve kuyruğu açmak için MQOPEN çağrısını kullanmanız (giriş, göz atma ya da her ikisi için) kullanmanız gerekir. Bu işlemler, “Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198 ve “Nesnelerin açılması ve kapatılması” sayfa 206içinde açıklanmaktadır.

Kuyruğu açtığınızda, aynı kuyruktaki iletilere göz atmak ya da iletileri kaldırmak için MQGET çağrısını sık olarak kullanabilirsiniz. Kuyruktan istediğiniz tüm iletileri almayı bitirdiğinizde, MQCLOSE ' yi çağırın.

Bir kuyruktan ileti alma hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “MQGET çağrısını kullanarak kuyruktan ileti alma” sayfa 231
- “İletilerin kuyruktan alınma sırası” sayfa 235
- “Belirli bir iletiyi alma” sayfa 246
- “Kalıcı olmayan iletilerin performansını artırma” sayfa 247
- “4 MB ' den büyük iletilerin işlenmesi” sayfa 251
- “İleti bekleniyor” sayfa 256
-
- “Geri alma işlemi atlanıyor” sayfa 256
- “Uygulama verileri dönüştürme” sayfa 259
- “Kuyruklardaki Iletilere Göz Atma” sayfa 260

- “MQGET çağrısının başarısız olduğu bazı durumlar” sayfa 265

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

MQGET çağrısını kullanarak kuyruktan ileti alma

MQGET çağrısı, açık yerel kuyruktan bir ileti alır. Başka bir sistemdeki kuyruktan ileti alamıyor.

MQGET çağrısına giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı.
- Kuyruk tanıtıcısı.
- Kuyruktan almak istediğiniz iletinin açıklaması. Bu, bir ileti tanımlayıcısı (MQMD) yapısıdır.
- İleti Alma Seçenekleri (MQGMO) yapısındaki bilgileri denetleyin.
- İletiyi tutmak için atadığınız arabelleğin büyüklüğü (MQlong).
- İletinin konulması için kullanılan depolama alanının adresi.

MQGET ' tan çıkış:

- Neden kodu
- Tamamlanma kodu
- Çağrı başarıyla tamamlanırsa, belirttiğiniz arabellek alanındaki ileti.
- Seçenek yapınız, iletinin alındığı kuyruğun adını göstermek için değiştirildi
- Alınan iletiyi açıklamak için değiştirilen alanların içeriğiyle ileti tanımlayıcı yapıınız
- İletinin uzunluğu (MQLONG)

MQGETiçinde MQGET çağrısının bir açıklaması var.

Aşağıdaki kısımlarda, MQGET çağrısına giriş olarak sağlamanız gereken bilgiler açıklanmaktadır.

- “Bağlantı tanıtıcılarının belirtilmesi” sayfa 232
- “MQMD yapısını ve MQGET çağrısını kullanarak iletileri tanımlama” sayfa 232
- “MQGMO yapısını kullanarak MQGET seçeneklerini belirtme” sayfa 232
- “Arabellek alanının büyüklüğünün belirtilmesi” sayfa 234

Bağlantı tanıtıcılarının belirtilmesi

z/OS üzerinde CICS uygulamaları için, MQHC_DEF_HCONN değişmezini (sıfır değerine sahip) belirtebilir ya da MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanabilirsiniz. Diğer uygulamalar için, her zaman MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını kullanın.

MQASAçık 'ı çağırdığınızda döndürülen kuyruk tanıtıcısını (*Hobj*) kullanın.

MQMD yapısını ve MQGET çağrısını kullanarak iletileri tanımlama

Bir kuyruktan almak istediğiniz iletiyi tanımlamak için, ileti tanımlayıcı yapısını (MQMD) kullanın.

Bu, MQGET çağrısına ilişkin bir giriş/çıkış parametresidir. MQMD 'nin “IBM WebSphere MQ ileti” sayfa 9' de tanımladığı ve MQMDiçinde yapının kendisiyle ilgili bir açıklaması olan ileti özelliklerine giriş var.

Kuyruktan hangi iletiyi almak istediğinizi biliyorsanız, bkz. “Belirli bir iletiyi alma” sayfa 246.

Belirli bir ileti belirtmezseniz, MQGET kuyrukta *ilk* iletisini alır. “İletilerin kuyruktan alınma sırası” sayfa 235 , bir iletinin önceliğinin, kuyruğun *MsgDeliverySequence* özniteliğinin ve MQGMO_LOGICAL_ORDER seçeneğinin, kuyruklardaki iletilerin sırasını nasıl belirlediğini açıklar.

Not: MQGET ' yi bir kereden fazla kullanmak istiyorsanız (örneğin, kuyrukta bulunan iletileri adımlamak için), her çağrıdan sonra bu yapının *MsgId* ve *CorrelId* alanlarını boş değer olarak ayarlamanız gerekir. Bu işlem, alınan iletinin tanıtıcılarının bu alanlarını temizler.

Ancak, iletilerinizi gruplamak istiyorsanız, *GroupId* ' un aynı gruptaki iletiler için aynı olması gerekir; böylece, arama, tüm grubu oluşturan bir iletinin önceki iletiyle aynı tanıtıcılara sahip olmasını sağlar.

MQGMO yapısını kullanarak MQGET seçeneklerini belirtme

MQGMO yapısı, MQGET çağrısına seçenekleri geçirmek için kullanılan bir giriş/çıkış değişkenidir. Aşağıdaki kısımlar, bu yapının bazı alanlarını tamamlamanıza yardımcı olur.

MQGMOiçinde MQGMO yapısının bir açıklaması vardır.

StrucId
StrucId , bir get-message options yapısı olarak yapıyı tanımlamak için kullanılan 4 karakterlik bir alandır. Her zaman MQGMO_STRUC_ID değerini belirtin.

Version
*Version*yapının sürüm numarasını açıklar. Varsayılan değerMQGMO_VERSION_1 ' dir. Sürüm 2 alanlarını kullanmak ya da iletileri mantıksal sırada almak istiyorsanız, MQGMO_VERSION_2değerini belirtin. Sürüm 3 alanlarını kullanmak ya da iletileri mantıksal sırayla almak istiyorsanız, MQGMO_VERSION_3değerini belirtin. MQGMO_CURRENT_VERSION uygulamanızı en son düzeyi kullanacak şekilde ayarlar.

Options
Kodunuz içinde, seçenekleri istediğiniz sırayla seçebilirsiniz; her seçenek *Options* alanında bir bit tarafından temsil edilir.

Options alan denetimleri:

- MQGET çağrısının, tamamlanmadan önce kuyruğa gelmesi için bekleyeceği (bkz. “İleti bekleniyor” sayfa 256)
- Alma işleminin bir iş birimine dahil edilip etmeyeceğini belirleyin.
- Kalıcı olmayan bir iletinin, eşitleme noktası dışında alınıp alınmadığı, hızlı ileti sistemine izin verilip verilmeyeceği
- On WebSphere MQ for z/OS, whether the message retrieved is marked as skipping backout (see “Geri alma işlemi atlanıyor” sayfa 256)
- İletinin kuyruktan kaldırılıp kaldırılmadığını ya da yalnızca göz atmadığını
- Göz atma ya da diğer seçim ölçütlerine göre bir iletinin seçilip seçilmeyeceği

- İleti arabelleğinizden daha uzun olsa bile çağrıların başarılı olup olmayacağını
- On WebSphere MQ for z/OS, whether to allow the call to complete. Bu seçenek, bir ileti geldiğinde bildirim almak istediğinizi belirtmek için bir sinyal de ayarlar.
- Kuyruk yöneticisi susturulmuş durumda olduğunda çağrıların başarısız olup olmadığını
- On WebSphere MQ for z/OS, whether the call fails if the connection is in a quiescing state
- Uygulama iletisi veri dönüştürmesinin gerekli olup olmadığı (bkz. “Uygulama verileri dönüştürme” sayfa 259)
- İletilerin ve (WebSphere MQ for z/OS) kesimlerinin bir kuyruktan alındığı sıralama düzeni
- Except on WebSphere MQ for z/OS, whether complete, logical messages only are retrievable
- Gruptaki iletilerin yalnızca, gruptaki *tüm* iletiler kullanılabildiği durumlarda alınıp alınmayacağı
- z/OSiçin WebSphere MQ dışında, mantıksal iletteki kesimler yalnızca mantıksal iletide *Tümü* bölümleri kullanılabilir olduğunda alınabilir.

Options alanını varsayılan değere (MQGMO_NO_WAIT) bırakırsanız, MQGET çağrısı şu şekilde çalışır:

- Kuyruktaki seçim ölçütlerinizle eşleşen bir ileti yoksa, çağrı ileti gelmesini beklemez, ancak hemen tamamlanır. Also, in WebSphere MQ for z/OS, the call does not set a signal requesting notification when such a message arrives.
- Arama, eşitleme noktalarıyla çalışma şeklinin altyapıya göre belirlendiği şekilde belirlenir:

Altyapı	**Eşitleme noktası denetimi altında**
IBM i	Hayır
UNIX and Linux sistemleri	Hayır
z/OS	Evet
Windows sistemleri	Hayır

- On WebSphere MQ for z/OS, the message retrieved is not marked as skipping backout.
- Seçilen ileti kuyruktan kaldırılır (göz atılmaz).
- Herhangi bir uygulama iletisi veri dönüştürme işlemi gerekmiyor.
- İleti arabelleğinizden uzunsa arama başarısız olur.

WaitInterval
WaitInterval alanı, MQGMO_WALEM seçeneğini kullandığınızda, MQGET çağrısının bir iletinin kuyruğa varması için bekleyeceği sürenin üst sınırını (milisaniye olarak) belirtir. *WaitInterval*' ta belirtilen süre içinde hiçbir ileti gelmezse, arama tamamlanır ve kuyrukta seçim ölçütlerinizle eşleşen bir ileti olmadığını gösteren bir neden kodu döndürür.

On WebSphere MQ for z/OS, if you use the MQGMO_SET_SIGNAL option, the *WaitInterval* field specifies the time for which the signal is set.

Bu seçenekler hakkında daha fazla bilgi için bkz. “İleti bekleniyor” sayfa 256.

Signal1
Signal1 is supported on WebSphere MQ for z/OS and MQSeries for HP Integrity NonStop Server only.

Uygun bir ileti geldiğinde uygulamanızın bilgilendirileceğini istemek için MQGMO_SET_SIGNAL seçeneğini kullanırsanız, *Signal1* alanında sinyal tipini belirtmiş olur. WebSphere MQ ' da diğer tüm platformlarda, *Signal1* alanı ayrılır ve değeri anlamlı değildir.

Signal2
Signal2 alanı tüm platformlarda ayrılır ve değeri önemli değildir.

ResolvedQName
ResolvedQName , kuyruk yöneticisinin iletinin alındığı kuyruğun adını (herhangi bir diğer ad çözüldükten sonra) döndürdüğü bir çıkış alanıdır.

MatchOptions
MatchOptions , MQGET için seçim ölçütlerini denetler.

GroupStatus
GroupStatus , aldığınız iletinin bir grup içinde olup olmadığını belirtir.

SegmentStatus
SegmentStatus , aldığınız öğenin mantıksal bir iletinin bir parçası olup olmadığını belirtir.

Segmentation
Segmentation , alınan ileti için bölümlemeye izin verilip verilmediğini belirtir.

MsgToken

MsgToken , bir iletiyi benzersiz şekilde tanımlar.

ReturnedLength
ReturnedLength , kuyruk yöneticisinin döndürdüğü ileti verilerinin uzunluğunu (bayt olarak) döndürdüğü bir çıkış alanıdır.

MsgHandle
Kuyruktan alınan iletinin özellikleri ile doldurulacak bir iletinin tanıtıcısı. Tanıtıcı daha önce bir MQCRTMH çağrısı tarafından yaratılmıştır. Bir ileti alınmadan önce, bu tanıtıcı ile ilişkilendirilmiş olan tüm özellikler temizlenir.

Arabellek alanının büyüklüğünün belirtilmesi

MQGET çağrısının *BufferLength* değiştirgesinde, aladığınız ileti verilerini tutmak için arabellek alanının büyüklüğünü belirtin. Bunun üç şekilde ne kadar büyük olması gerektiğine karar verirsiniz:

1. Bu programdan beklenecek iletilerin uzunluğunu zaten bilebilirsiniz. Bu durumda, bu boyutta bir arabellek belirtin.

   Ancak, ileti arabellek için çok büyük olsa da MQGET çağrısının tamamlanmasını istiyorsanız, MQGMO yapısındaki MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini kullanabilirsiniz. Bu durumda:

   - Arabellek, tutulabildiği kadar iletiyi doldurur.
   - Çağrı, bir uyarı tamamlanma kodu döndürür
   - İleti kuyruktan kaldırılır (iletinin geri kalan kısmı atılır) ya da göz at imleci ilerletilir (kuyruğa göz atıyorsanız).
   - İletinin gerçek uzunluğu *DataLength*içinde döndürülür.

   Bu seçenek olmadan, çağrı bir uyarıyla tamamlanır, ancak iletiyi kuyruktan kaldırmaz (ya da göz atma imlecini ilerletmez).
2. Arabellek için bir boyut tahmin edin (ya da sıfır byte büyüklüğünde bir boyut belirtin) ve *yapma* , MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini kullanın. MQGET çağrısının başarısız olması (örneğin, arabellek çok küçük olduğu için), iletinin uzunluğu çağrıya ilişkin *DataLength* parametresine döndürülür. (Arabellek, iletinin tutulabildiği kadarını doldurur, ancak arama işlemi tamamlanmamaktadır.) Bu iletinin *MsgId* değerini saklayın, daha sonra, doğru büyüklükte bir arabellek alanı belirterek, MQGET çağrısını yineleyin ve ilk çağrıdan not ettiğiniz *MsgId* ' i yineleyin.

   Programınız başka programlar tarafından da sunulmakta olan bir kuyruğa sunuyorsa, diğer programlardan biri, programınız başka bir MQGET çağrısını yayınlamadan önce istediğiniz iletiyi kaldırabilir. Programınız, artık var olmayan bir iletiyi arayarak zaman kaybedebilir. Bunu önlemek için, istediğiniz iletiyi buluncaya kadar önce kuyruğa göz atın, sıfır *BufferLength* belirterek ve MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini kullanın. Bu, göz atma imlecini istediğiniz iletinin altına konumlayın. Bundan sonra MQGET çağrılarak, MQGMO_MSG_UNDER_CURSOR seçeneğini belirterek iletiyi alabilirsiniz. Göz atma ve kaldırma çağrılarınız arasındaki iletiyi başka bir program

kaldırırsa, göz atma imleciniz altında ileti olmadığı için, ikinci MQGET işlemi hemen başarısız olur (tüm kuyruğun aranması olmadan).

3. *MaxMsgLength queue* özniteliği, o kuyruk için kabul edilen ileti uzunluğu üst sınırını belirler; *MaxMsgLength kuyruk yöneticisi* özniteliği, o kuyruk yöneticisi için kabul edilen ileti uzunluğu üst sınırını belirler. Beklenecek iletinin uzunluğunu bilmiyorsanız, *MaxMsgLength* özniteliğini sorgulayabilir (MQINQ çağrısını kullanarak), bu büyüklükte bir arabellek belirtebilirsiniz.

   Başarımı düşürmemek için arabellek büyüklüğünü, gerçek ileti büyüklüğünün olabildiğince yakın olması için deneyin.

   MaxMsgLength özniteliğe ilişkin daha fazla bilgi için bkz. "İleti uzunluğu üst sınırını artırma" sayfa 251.

İletilerin kuyruktan alınma sırası

Kuyruktan ileti almanıza yardımcı olacak sırayı denetleyebilirsiniz. Bu bölüm seçeneklere bakar.

Öncelik

Bir program, iletiyi bir kuyruğa koyduğunda bir ileti için öncelik atayabilir (bkz. "İleti öncelikleri" sayfa 17). Eşit önceliğe sahip iletiler, işlendikleri sıraya göre değil, geliş sırasına göre kuyrukta saklanır.

Kuyruk yöneticisi, kuyrukları, sıkı FIFO (ilk giren, ilk çıkış) sırasıyla ya da öncelik sırası içinde FIFO ' da tutar. Bu, kuyruğun *MsgDeliverySequence* özniteliğinin ayarına bağlıdır.Bir ileti kuyruğa ulaştığında, aynı önceliğe sahip son iletinin hemen ardından eklenir.

Programlar bir kuyruktan ilk iletiyi alabilir ya da bir kuyruktan belirli bir iletiyi alabilirler, bu iletilerin önceliğini dikkate almayabilir. Örneğin, bir program, yanıtı daha önce gönderdiği belirli bir iletiye işlemek isteyebilirler. Daha fazla bilgi için "Belirli bir iletiyi alma" sayfa 246 başlıklı konuya bakın.

Bir uygulama kuyrukta bir ileti dizisi koyarsa, başka bir uygulama bu iletileri, yerleştirdikleri sırayla alabilir ve bu iletileri aşağıdaki sırayla alabilir:

- İletilerin hepsinin önceliği aynı.
- Mesajların hepsi aynı iş birimi içinde, ya da hepsi bir iş biriminin dışına konulmak üzere.
- Kuyruk, uygulama koymak için yereldir

Bu koşullar karşılanmazsa ve uygulamalar belirli bir sırayla alınmakta olan iletilere bağlı olduğunda, uygulamalar ileti verilerinde sıralama bilgilerini içermeli ya da bir iletinin sonraki gönderilmeden önce bir iletinin alınmasını kabul etmek için bir araç oluşturmalıdır.

Mantıksal ve fiziksel sıralama

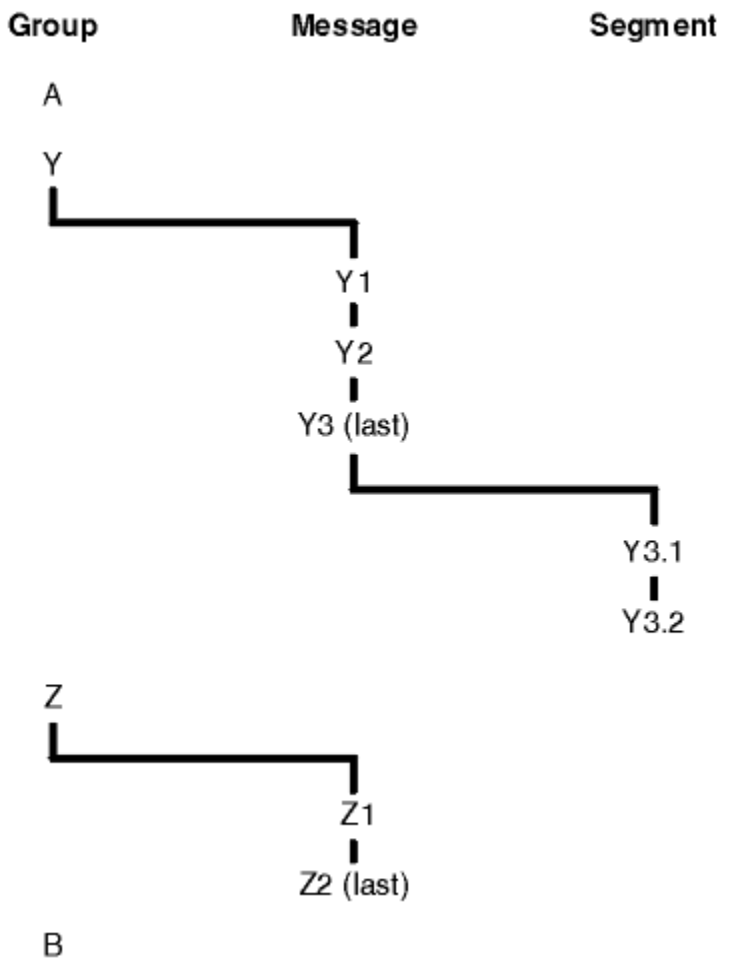
Kuyruklardaki iletiler *fiziksel* ya da *mantıksal* düzende (her bir öncelik düzeyi içinde) gerçekleşebilir.

Fiziksel sıralama, iletilerin kuyrukta vardığı sıradır. Mantıksal sipariş, bir gruptaki tüm ileti ve kesimlerin, gruba ait ilk öğenin fiziksel konumu tarafından belirlenen konumda, birbirinin yanında mantıksal sırada yer aldıklarında yer alan bir sıradır.

Grupların, iletilerin ve bölümlerin bir açıklaması için bkz. "İleti grupları" sayfa 34. Bu fiziksel ve mantıksal siparişler farklı olabilir:

- Gruplar, farklı uygulamalardan benzer zamanlarda bir hedefe varabilir ve bu nedenle farklı fiziksel düzeni kaybedebilir.
- Tek bir grup içinde bile iletiler, gruptaki iletilerin bir kısmının yeniden yönlendirmesi ya da gecikmesi nedeniyle sıradan çıkabiliyor.

Örneğin, mantıksal sipariş Şekil Şekil 34 sayfa 236gibi görünebilir:

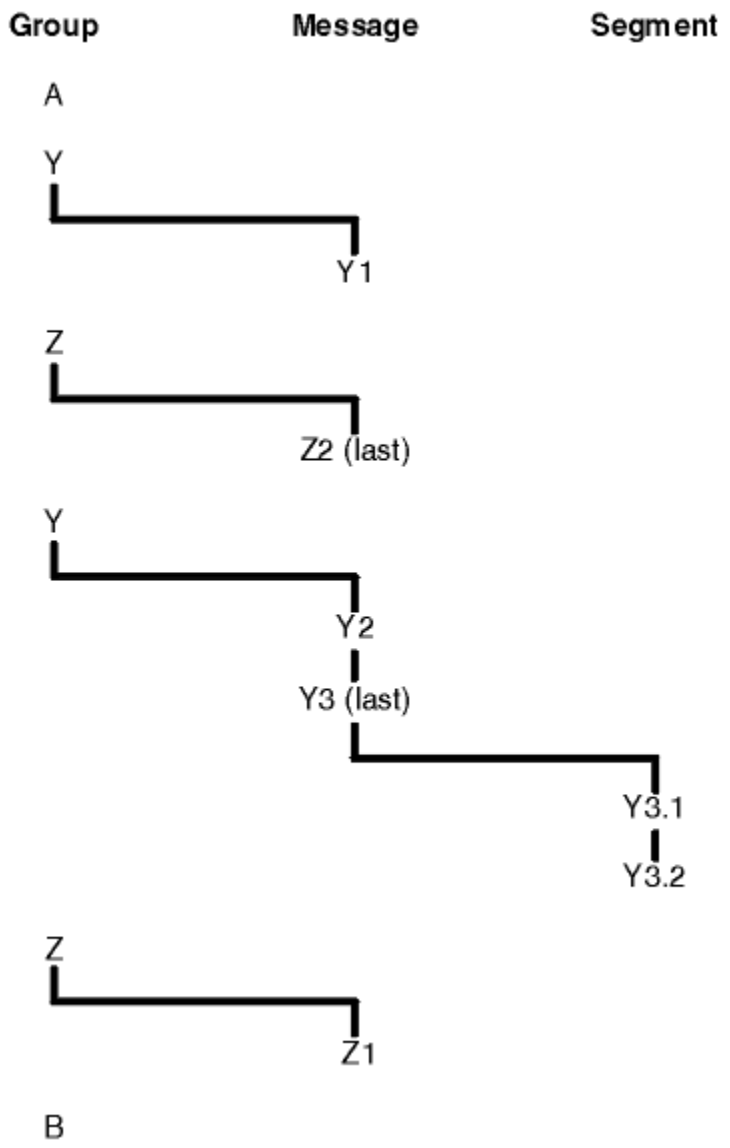


Şekil 34. Kuyruğun mantıksal sırası

Bu iletiler, bir kuyrukta aşağıdaki mantıksal sırada gerçekleşir:

1. İleti A (bir grupta değil)
2. Grup Y ' nin mantıksal iletisi 1
3. Grup Y ' nin mantıksal ileti 2
4. Grup Y ' nin (son) mantıksal ileti 3 'ünün (son) 1. bölümü
5. (Son) grup Y ' nin 2 numaralı (son) mantıksal iletisi 3
6. Grup Z ' nin mantıksal iletisi 1
7. (Son) grup Z ' nin mantıksal iletisi 2
8. İleti B (bir grupta değil)

Ancak fiziksel düzen tamamen farklı olabilir. Her bir grup içindeki *ilk* öğenin fiziksel konumu, tüm grubun mantıksal konumunu belirler. Örneğin, Y ve Z grupları benzer zamanlarda geldiyse ve Z grup Z ' nin iletisi aynı grubun 1 numaralı iletisini aşıyorsa, fiziksel sipariş Şekil Şekil 35 sayfa 237gibi görünür:



Şekil 35. Kuyruklardaki fiziksel sıralama

Bu iletiler, kuyrukta aşağıdaki fiziksel sırada gerçekleşir:

1. İleti A (bir grupta değil)
2. Grup Y ' nin mantıksal iletisi 1
3. Z grubu Z mantıksal iletisi 2
4. Grup Y ' nin mantıksal ileti 2
5. Grup Y ' nin (son) mantıksal ileti 3 'ünün (son) 1. bölümü
6. (Son) grup Y ' nin 2 numaralı (son) mantıksal iletisi 3
7. Grup Z ' nin mantıksal iletisi 1
8. İleti B (bir grupta değil)

Not: IBM WebSphere MQ for z/OS' ta, kuyruğun GROUPID tarafından dizinlenmesi durumunda, kuyruklardaki iletilerin fiziksel sırası garanti edilmez.

İletileri alırken, iletileri fiziksel sıralama yerine mantıksal sırada almak için MQGMO_LOGICAL_ORDER belirtebilirsiniz.

MQGMO_BROWSE_FIRST ve MQGMO_LOGICAL_ORDER ile MQGET işlemi yayınlarsa, MQGMO_BROWSE_NEXT ile sonraki MQGET çağrıları da MQGMO_LOGICAL_ORDER değerini de

belirtmelidir. Bunun tersine, MQGMO_Browse_first ile MQGET işlemi MQGMO_LOGICAL_ORDER belirtmiyorsa, MQGMO_BROWSE_NEXT ile birlikte izleyen MQGES ' ler de geçerli değildir.

Kuyruk yöneticisinin MQGET çağrıları için sakladığı grup ve kesim bilgileri, kuyruklardaki iletilere göz atmanın grup ve bölüm bilgilerinden ayrıdır ve kuyruk yöneticisinin iletileri kuyruktan kaldırmak için MQGET çağrıları için sakladığı bilgileri içerir. MQGMO_BROWSE_FIRST belirttiğinizde, kuyruk yöneticisi, göz atma için grup ve bölüm bilgilerini yoksayar ve yürürlükteki grup ve yürürlükteki mantıksal ileti yok gibi kuyrukları tarar.

Not: Do not use an MQGET call to browse *sonun ötesinde* of a message group (or logical message not in a group) without specifying MQGMO_LOGICAL_ORDER. For example, if the last message in the group *emsaller* the first message in the group on the queue, using MQGMO_BROWSE_NEXT to browse beyond the end of the group, specifying MQMO_MATCH_MSG_SEQ_NUMBER with *MsgSeqNumber* set to 1 (to find the first message of the next group) returns again the first message in the group already browsed. Bu durum hemen olabilir ya da daha sonra (araya giren gruplar varsa) MQGET çağrıları sayısı hemen olabilir.

Göz atma işlemi için *iki kez* kuyruğunu açarak sonsuz döngü olasılığını önlein:

- Her gruptaki ilk iletiye göz atmak için ilk tanıtıcıyı kullanın.
- Yalnızca belirli bir grup içindeki iletilere göz atmak için ikinci tanıtıcıyı kullanın.
- Gruptaki iletilere göz atmadan önce, ikinci göz atma imlecini ilk göz atma imlecinin konumuna taşımak için MQMO_ * seçeneklerini kullanın.
- Bir grubun sonuna kadar MQGMO_BROWSE_NEXT göz atma olanağını kullanmayın.

Bu konuda ek bilgi için bkz. MQGET, MQMDve MQI seçeneklerinin geçerliliğini denetlemek için kurallar.

Çoğu uygulama için, göz atılırken mantıksal ya da fiziksel sıralamayı seçebilirsiniz. Ancak, bu kipler arasında geçiş yapmak istiyorsanız, ilk olarak MQGMO_LOGICAL_ORDER ile bir göz atma işlemini ilk kez yayınlarken, mantıksal sıra içindeki konumunuz oluşturulur.

Gruptaki ilk öğe şu anda mevcut değilse, içinde bulunmanız gereken grup, mantıksal sıranın bir parçası olarak kabul edilmez.

Göz at imleci bir grup içindeyse, ilk ileti kaldırılsa bile, aynı grup içinde devam edebilir. Başlangıçta, ilk öğenin mevcut olmadığı MQGMO_LOGICAL_ORDER kullanarak bir gruba hiçbir zaman geçemeyebilirsiniz.

MQPMO_LOGICAL_ORDER

MQPMO seçeneği, kuyruk yöneticisine uygulamanın, iletileri gruplar ve mantıksal iletiler segmentlerine nasıl yerleştirdiğini bildirir. Yalnızca MQPUT çağrısında belirtilebilir; MQPUT1 çağrısında geçerli değildir.

MQPMO_LOGICAL_ORDER belirtildiyse, uygulamanın art arda gelen MQPUT çağrılarını kullandığını gösterir:

1. Her bir mantıksal iletiye, boşluk olmadan, 0 'dan başlayarak, artan kesim göreli konumu sırasına göre kesimler yerleştirin.
2. Bölümleri sonraki mantıksal iletiye koymadan önce, tüm bölümleri bir mantıksal iletiye koyun.
3. Herhangi bir boşluk olmadan, 1 'den başlayarak, ileti sıra sayısı artırımı sırasına göre, her ileti grubuna mantıksal iletileri yerleştirin. IBM WebSphere MQ otomatik olarak ileti sıra numarasını artırır.
4. Sonraki ileti grubuna mantıksal iletiler koymadan önce, tüm mantıksal iletileri bir ileti grubuna koyun.

Uygulama kuyruk yöneticisine, iletileri gruplar ve mantıksal iletiler bölümlerine nasıl yerleştirdiğini anlattığından, kuyruk yöneticisi bu bilgileri saklayıp güncellediğinden, uygulamanın her bir MQPUT çağrısıyla ilgili grup ve bölüm bilgilerini korumak ve güncellemek zorunda kalmayacağından emin olun. Özellikle, kuyruk yöneticisi bu alanları uygun değerlere ayarlandığından, uygulamanın MQMD ' de *GroupId*, *MsgSeqNumber*ve *Offset* alanlarını ayarlamaya gerek olmadığı anlamına gelir. Uygulamanın yalnızca MQMD ' deki *MsgFlags* alanını ayarlaması gerekir; bu alan, iletilerin ne zaman gruplara ait olduğunu ya da mantıksal iletilerin bölümleri olduğunu göstermek ve bir gruptaki son iletiyi ya da mantıksal iletinin son bölümünü belirtmek için.

After a message group or logical message has been started, subsequent MQPUT calls must specify the appropriate MQMF_* flags in *MsgFlags* in MQMD. Uygulama, sonlandırılmamış bir ileti grubu olduğunda ya da sonlandırılmamış bir mantıksal ileti olduğunda, bir ileti grubu olmayan bir iletiyi yerleştirmeye çalışırsa, uygun olduğu şekilde, çağrı neden kodu MQRC_INCOMPLE_GROUP ya da MQRC_INCOMPLE_MSG neden koduyla başarısız olur. Ancak, kuyruk yöneticisi yürürlükteki ileti grubuyla ya da yürürlükteki mantıksal iletiyle ilgili bilgileri saklar ve MQMF_LAST_MSG_IN_GROUP ya da MQMF_LAST_SEGMENT belirtimini uygun olarak belirterek, MQPUT çağrısını, grupta olmayan ya da bir kesim olmayan iletiyi koymak için yeniden vermeden önce, uygulama bu bilgileri bir ileti (uygulama iletisi verisi olmadan) göndererek sonlandırabilir.

Şekil 35 sayfa 237 , geçerli olan seçenek ve işaretlerin birleşimlerini ve kuyruk yöneticisinin her bir durumda kullandığı *GroupId*, *MsgSeqNumber*ve *Offset* alanlarının değerlerini gösterir. Tabloda gösterilmeyen seçenek ve işaret birleşimleri geçerli değil. Çizelgedeki kolonlar aşağıdaki anlamlara sahiptir; Evet ya da Hayır değeri anlamına gelir:

GüNLüK ORD
Çağrıda MQPMO_LOGICAL_ORDER seçeneğinin belirtilip belirtilmediğini belirleyin.

MIG
Çağrıda MQMF_MSG_IN_GROUP ya da MQMF_LAST_MSG_IN_GROUP seçeneğinin belirtilip belirtilmediğini.

SEG
Çağrıda MQMF_SEGMENT ya da MQMF_LAST_SEGMENT seçeneğinin belirtilip belirtilmediğini belirleyin.

SEç TAMAM
Çağrıda MQMF_SEGMENTATION_ALLOWLI seçeneğinin belirtilip belirtilmediğini.

Cur Grp
Çağrıdan önce geçerli bir ileti grubunun var olup olmadığını.

Ccur günlük iletisi
Çağrıdan önce geçerli bir mantıksal iletinin var olup olmadığı.

Diğer kolonlar
Kuyruk yöneticisinin kullandığı değerleri gösterir. Önceki ileti, kuyruk tanıtıcısı için önceki iletide alan için kullanılan değeri gösterir.

Çizelge 36. Mantıksal iletilerin gruplarındaki ve kısımlarındaki iletilere ilişkin MQPUT seçenekleri

Belirttiğiniz seçenekler				**Aramadan önce grup ve günlük-msg durumu**		**Kuyruk yöneticisinin kullandığı değerler**		
OTURUM KAPAT	**MIG**	**GÇ**	**SEEG Tamam**	**Cur grp**	**Cur günlük iletisi**	***GroupId***	***MsgSeqNumber***	***Offset***
Evet	Hayır	Hayır	Hayır	Hayır	Hayır	MQGI_NONE	1	0
Evet	Hayır	Hayır	Evet	Hayır	Hayır	Yeni grup tanıtıcısı	1	0
Evet	Hayır	Evet	Herhangi biri	Hayır	Hayır	Yeni grup tanıtıcısı	1	0
Evet	Hayır	Evet	Herhangi biri	Hayır	Evet	Önceki grup tanıtıcısı	1	Önceki göreli konum + önceki bölüm uzunluğu
Evet	Evet	Herhangi biri	Herhangi biri	Hayır	Hayır	Yeni grup tanıtıcısı	1	0

Çizelge 36. Mantıksal iletilerin gruplarındaki ve kısımlarındaki iletilere ilişkin MQPUT seçenekleri (devamı var)

Belirttiğiniz seçenekler				Aramadan önce grup ve günlük-msg durumu		Kuyruk yöneticisinin kullandığı değerler		
OTURUM KAPAT	**MIG**	**GÇ**	**SEEG Tamam**	**Cur grp**	**Cur günlük iletisi**	***GroupId***	***MsgSeqNumber***	***Offset***
Evet	Evet	Herhangi biri	Herhangi biri	Evet	Hayır	Önceki grup tanıtıcısı	Önceki sıra numarası + 1	0
Evet	Evet	Evet	Herhangi biri	Evet	Evet	Önceki grup tanıtıcısı	Önceki sıra numarası	Önceki göreli konum + önceki bölüm uzunluğu
Hayır	Hayır	Hayır	Hayır	Herhangi biri	Herhangi biri	MQGI_NONE	1	0
Hayır	Hayır	Hayır	Evet	Herhangi biri	Herhangi biri	MQGI_NONE ise, alanda başka değer varsa yeni grup tanıtıcısı	1	0
Hayır	Hayır	Evet	Herhangi biri	Herhangi biri	Herhangi biri	MQGI_NONE ise, alanda başka değer varsa yeni grup tanıtıcısı	1	Alandaki değer
Hayır	Evet	Hayır	Herhangi biri	Herhangi biri	Herhangi biri	MQGI_NONE ise, alanda başka değer varsa yeni grup tanıtıcısı	Alandaki değer	0
Hayır	Evet	Evet	Herhangi biri	Herhangi biri	Herhangi biri	MQGI_NONE ise, alanda başka değer varsa yeni grup tanıtıcısı	Alandaki değer	Alandaki değer

Not:

- MQPMO_LOGICAL_ORDER, MQPUT1 çağrısında geçerli değil.
- *MsgId* alanı için, MQPMO_NEW_MSG_ID ya da MQMI_NONE belirtilirse, kuyruk yöneticisi yeni bir ileti tanıtıcısı oluşturur ve bu değeri alanda başka bir değer kullanır.
- *CorrelId* alanı için, MQPMO_NEW_CORREL_ID belirtildiyse, kuyruk yöneticisi yeni bir ilinti tanıtıcısı oluşturur ve diğer bir biçimde alanda değer kullanır.

MQPMO_LOGICAL_ORDER belirttiğinizde, kuyruk yöneticisi bir gruptaki tüm iletilerin ve mantıksal bir iletide belirtilen tüm iletilerin, MQMD ' deki *Persistence* alanında aynı değere sahip olmasını gerektirir; yani, tüm bunların kalıcı olması ya da tümünün kalıcı olmaması gerekir. Bu koşul karşılanmazsa, MQPUT çağrısına MQRC_INCONSISTENT_PERSISTENCE neden koduyla başarısız olur.

MQPMO_LOGICAL_ORDER seçeneği, iş birimlerini aşağıdaki gibi etkiler:

- Bir gruptaki ilk fiziksel ileti ya da mantıksal ileti bir iş birimi içine konursa, aynı kuyruk tanıtıcısı kullanılsa, gruptaki ya da mantıksal iletteki diğer tüm fiziksel iletiler bir iş birimi içine konmalıdır. Ancak, iki ya da daha fazla sayıda fiziksel iletilerden oluşan bir ileti grubunun ya da mantıksal iletinin kuyruk tanıtıcısı için birbirini izleyen iki ya da daha çok iş birimi arasında bölünmesine olanak sağlayan bir ileti grubuna ya da mantıksal iletiye izin verilmesine gerek yoktur.

- Bir gruptaki ilk fiziksel ileti ya da mantıksal ileti bir iş birimi içine konmazsa, aynı kuyruk tanıtıcısı kullanılsa, gruptaki diğer fiziksel iletilerden hiçbiri ya da mantıksal ileti bir iş birimi içine yerleştirilebilir.

Bu koşullar karşılanmazsa, MQPUT çağrısına neden kodu MQRC_INCONSISTENT_UOW neden kodu girilir.

MQPMO_LOGICAL_ORDER belirtildiğinde, MQPUT çağrısında belirtilen MQMD, MQMD_VERSION_2değerinden küçük olmamalıdır. Bu koşul karşılanmazsa, çağrı neden kodu MQRC_WRONG_MD_VERSION ile başarısız olur.

MQPMO_LOGICAL_ORDER belirtilmezse, mantıksal iletilerin gruplarındaki ve kısımlarındaki iletiler herhangi bir sıraya yerleştirilebilir ve tam ileti gruplarının ya da tam mantıksal iletilerin tamamlanmaması gerekmez. It is the responsibility of the application to ensure that the *GroupId*, *MsgSeqNumber*, *Offset*, and *MsgFlags* fields have appropriate values.

Bir sistem hatası ortaya çıktıktan sonra, ortadaki bir ileti grubunu ya da mantıksal iletiyi yeniden başlatmak için bu tekniği kullanın. Sistem yeniden başlatıldığında, uygulama *GroupId*, *MsgSeqNumber*, *Offset*, *MsgFlags*ve *Persistence* alanlarını uygun değerlere ayarlayabilir ve daha sonra MQPMO_SYNCPNODER ya da MQPMO_NO_SYNCPOINT ayarlı MQPUT çağrısını zorunlu olarak ayarlayabilir, ancak MQPMO_LOGICAL_ORDER belirtilmeden. Bu çağrı başarılı olursa, kuyruk yöneticisi grup ve bölüm bilgilerini saklar ve izleyen MQPUT çağrıları bu kuyruk tanıtıcısını kullanarak MQPMO_LOGICAL_ORDER değerini normal olarak belirtebilir.

Kuyruk yöneticisinin MQPUT çağrısı için sakladığı grup ve kesim bilgileri, MQGET çağrısına ilişkin sakladığı grup ve bölüm bilgilerinden ayrıdır.

Herhangi bir kuyruk tanıtıcısı için uygulama, MQPMO_LOGICAL_ORDER ile belirtilen MQPUT çağrılarını karıştırabilir; bu çağrılar, aşağıdaki noktaları dikkate almaz:

- MQPMO_LOGICAL_ORDER belirtilmediyse, her başarılı MQPUT çağrısı kuyruk yöneticisinin, kuyruk tanıtıcısı için kuyruk yöneticisi tarafından tutulan var olan grup ve kesim bilgilerinin yerine, kuyruk tanıtıcısı için grup ve kesim bilgilerini uygulama tarafından belirlenen değerlere ayarlamasına neden olur.
- MQPMO_LOGICAL_ORDER belirtilmediyse, yürürlükteki ileti grubu ya da mantıksal ileti varsa arama başarısız olmaz; çağrıya MQCC_UYARI tamamlanma kodu ile başarılı olabilir. Çizelge 37 sayfa 241 ortaya çıkabilecek çeşitli vakaları gösterir. Bu durumlarda, tamamlanma kodu MQCC_OK değilse, neden kodu aşağıdakilerden biridir (uygun olduğu gibi):
  - MQRC_INCOMPLE_GROUP
  - MQRC_INCOMPLE_MSG
  - MQRC_INTUTARLMENT_PERSISTENCE
  - MQRC_INCONSISTENT_UOW

  Not: Kuyruk yöneticisi, MQPUT1 çağrısına ilişkin grup ve bölüm bilgilerini denetlemez.

Çizelge 37. MQPUT ya da MQCLOSE çağrısının grup ve bölüm bilgileriyle tutarlı olmadığı bir sonuç

Yürürlükteki çağrı	Önceki arama MQPMO_LOGICAL_ORDER ile MQPUT oldu	Önceki arama MQPMO_LOGICAL_ORDER olmadan MQPUT oldu
MQPUT ile MQPMO_LOGICAL_ORDER	MQCC_FAILED	MQCC_FAILED
MQPMO_LOGICAL_ORDER olmadan MQPUT	MQCC_UYARı	MQCC_OK
Sonlandırılmamış bir grupla ya da mantıksal iletiyle MQCLOSE	MQCC_UYARı	MQCC_OK

İletileri ve bölümleri mantıksal sırada koyan uygulamalar için, kullanılacak en basit seçenek olduğu için MQPMO_LOGICAL_ORDER değerini belirtin. Bu seçenek, kuyruk yöneticisi bu bilgileri yönettiği için, grup ve bölüm bilgilerini yönetme gereksiniminin uygulanını giderir. Ancak, özelleştirilmiş uygulamaların MQPMO_LOGICAL_ORDER seçeneği tarafından sağlanandan daha fazla denetime gereksinimi olabilir. Bu durumda, bu seçeneğin belirlenmemesi sağlanabilir; bunu gerçekleştirdiğinizde, MQMD ' deki*GroupId*, *MsgSeqNumber*, *Offset*ve *MsgFlags* alanlarının her bir MQPUT ya da MQPUT1 çağrısından önce doğru olarak ayarlandığından emin olmanız gerekir.

Örneğin, bu iletilerin grup halinde mi, yoksa mantıksal ileti bölümleri mi olduğunu dikkate almadan, aldığı fiziksel iletileri iletmek isteyen bir uygulama, iki nedenden dolayı MQPMO_LOGICAL_ORDER belirtmemelidir:

- İletiler alındıysa ve sıraya konursa, MQPMO_LOGICAL_ORDER belirtilirse, iletilere yeni bir grup tanıtıcısı atar; bu, iletilerin kaynağı için ileti grubunun sonucundaki yanıt ya da rapor iletilerini ilintilendirmek için zorlanabilir ya da imkansız hale gelebilir.
- Kuyruk yöneticileri gönderme ve alma arasında birden çok yol içeren karmaşık bir ağda, fiziksel iletiler sıradan çıkabilirler. MQGET çağrısında MQPMO_LOGICAL_ORDER ve MQGMO_LOGICAL_ORDER belirtilmeden, iletme uygulaması her fiziksel iletiyi alır almaz ve mantıksal sırada gelecek mantıksal sırada beklemeden alabilir ve iletebilirler.

Grup ya da mantıksal ileti bölümlerindeki iletiler için rapor iletileri oluşturan uygulamaların, rapor iletisi yerleştirilirken MQPMO_LOGICAL_ORDER belirtmemesi de gerekir.

MQPMO_LOGICAL_ORDER, diğer MQPMO_ * seçeneklerinin hiçbiriyle belirtilebilir.

Mantıksal Olarak Sıralı Grupları Kümelenmiş Bir Kuyruğa Koyma (MQOO_BIND_ON_GROUP)

MQOO_BIND_ON_OPEN seçeneği, bu uygulamadaki tüm iletilerin ve dolayısıyla tüm grupların tek bir yönetim ortamına yöneltilmesini sağlar. Bu, uygulama trafiğinin bir küme kuyruğunun birden çok örneğinde dengeli olarak yüklenmemesi için bir dezavantasa sahiptir. İş yükü dengelemeyi etkin bir şekilde tutarken, iş yükü dengelemeyi etkinleştirmek için aşağıdaki seçenekleri ayarlamanız gerekir:

- MQPUT çağrısının MQPMO_LOGICAL_ORDER belirtilmeli
- MQOPEN çağrısı aşağıdaki iki seçenekten birini belirtmelidir:
  - MQOO_BIND_ON_GROUP
  - MQOO_BIND_AS_Q_DEF ve kuyruk tanımlamasının DEFBIND (GROUP) belirtmesi gerekir

İş yükü dengelemesi, kuyruğun MQCLOSE ve MQOPER gerektirmeden, iletilerin *gruplar arasında* yönlendirilmesini sağlar. *Gruplar arasında* , MQMF_MSG_IN_GROUP ' un MQMD (v2) ya da MQMDE olarak ayarının olduğu ve devam etmekte olan bir kısmı tam grubu olmadığı anlamına gelir. Bir grup devam ederken, nesne tutamacındaki çözülen kuyruk yöneticisi ve kuyruk adı yeniden kullanılır.

Önceki ileti MQPMO_LOGICAL_ORDER ve/ya da MQMF_MSG_IN_GROUP ise, ancak yürürlükteki ileti grubun bir parçası değil, PUT çağrısı MQRC_INCOMPLE_GROUPile başarısız olur.

Tek bir MQPUT işlevi MQPMO_LOGICAL_ORDER belirtmezse ve yürürlükteki grup etkin değilse, o ileti için iş yükü dengelemesi sürülüyorsa (bu MQOPEN çağrısının MQOO_BIND_NOT_FIXED belirtmiş gibi).

MQOO_BIND_ON_GROUP kullanan bir hedefe yönelik iletiler için gerçek konum yok. Gerçek konum hakkında daha fazla bilgi için bkz. “İleti grupları” sayfa 34.

Mantıksal iletileri gruplama

Bir gruptaki mantıksal iletilerin kullanılmasının başlıca iki nedeni vardır:

- İletileri belirli bir sırada işlemeniz gerekebilir.
- Bir gruptaki her bir iletiyi ilgili bir şekilde işlemeniz gerekebilir.

Her iki durumda da, aynı uygulama örneğine sahip tüm grubu alın.

Örneğin, grubun dört mantıksal iletinden oluştuğunu varsayın. Uygulama koyma işlemi şu şekilde görünüyor:

```
PMO.Options = MQPMO_LOGICAL_ORDER | MQPMO_SYNCPOINT

MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_LAST_MSG_IN_GROUP

MQCMIT
```

Uygulama alma işlemi, gruptaki ilk ileti için MQGMO_ALL_MSGS_AVAM seçeneğini belirtir. Bu, grup içindeki tüm iletiler ulaşmadan işlemin başlatılmamasını sağlar. Grup içindeki sonraki iletiler için MQGMO_ALL_MSGS_AVAILD seçeneği yoksayıldı.

Grubun ilk mantıksal iletisi alındığında, grubun geri kalan mantıksal iletilerinin sırayla alındığından emin olmak için MQGMO_LOGICAL_ORDER seçeneğini kullanabilirsiniz.

Yani, uygulama alma şu şekilde görünüyor:

```
/* Wait for the first message in a group, or a message not in a group */
GMO.Options = MQGMO_SYNCPOINT | MQGMO_WAIT
            | MQGMO_ALL_MSGS_AVAILABLE | MQGMO_LOGICAL_ORDER
do while ( GroupStatus == MQGS_MSG_IN_GROUP )
   MQGET
   /* Process each remaining message in the group */
   ...

MQCMIT
```

İletileri gruplamak için daha fazla örnek için bkz. “Mantıksal iletilerin uygulama bölümlemesi” sayfa 253 ve “İş birimlerine yayılan bir grubu koymak ve almak” sayfa 243.

Bir uygulamanın, küme kuyrukları için aynı hedef somut örneğe ayrılmış bir ileti grubunu istemesine izin verilmesine ilişkin bilgi için DefBindbaşlıklı konuya bakın.

İş birimlerine yayılan bir grubu koymak ve almak

Önceki durumda, iletiler ya da kesimler düğümden ayrılmadan (hedef uzaksa) ya da tüm grup çalışmaya başlayıncaya ve iş birimi kesinleştirilinceye kadar, alınmaya başlayamaz. Bu, tüm grubu koymak uzun sürerse ya da düğüm üzerinde kuyruk alanı sınırlıysa, istediğiniz bu olmayabilir. Bunu aşmak için, grubu birkaç ünite işe sokun.

Grup birden çok iş birimi içine yerleştirilirse, uygulama koyma işlemi başarısız olduğunda da bazı grup tarafından kesinleştirilmesinin mümkün olur. Bu nedenle, uygulama, tamamlanmamış bir grubu sürdürmek için yeniden başlatma işleminden sonra kullanabileceği, her iş birimi ile kesinleştirilen durum bilgilerini kaydetmelidir. Bu bilgileri kaydetmenin en basit yeri STATUS kuyruğunda yer alıyor. Tam bir grup başarıyla ortaya konulduysa, STATUS kuyruğu boş olur.

Bölümleme ilişkisine dahil olursa, mantık benzerdir. Bu durumda, StatusInfo ' nın *Offset* içermesi gerekir.

Burada, grubu birkaç iş birimine yerleştirmenin bir örneği yer almaktadır:

```
PMO.Options = MQPMO_LOGICAL_ORDER | MQPMO_SYNCPOINT

/* First UOW */

MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
StatusInfo = GroupId,MsgSeqNumber from MQMD
MQPUT (StatusInfo to STATUS queue) PMO.Options = MQPMO_SYNCPOINT
MQCMIT

/* Next and subsequent UOWs */
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
```

```
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQGET (from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
StatusInfo = GroupId,MsgSeqNumber from MQMD
MQPUT (StatusInfo to STATUS queue) PMO.Options = MQPMO_SYNCPOINT
MQCMIT

/* Last UOW */
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
MQPUT MD.MsgFlags = MQMF_LAST_MSG_IN_GROUP
MQGET (from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
MQCMIT
```

Tüm iş birimleri kesinleştirildiyse, tüm grup başarıyla yerleştirilir ve STATUS kuyruğu boş olur. Aksi takdirde, grubun durum bilgisiyle belirtilen noktada sürdürülmesi gerekir. MQPMO_LOGICAL_ORDER ilk kez koyma için kullanılamaz, ancak bundan sonra olabilir.

Yeniden başlatma işlemi şu şekilde görünür:

```
MQGET (StatusInfo from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
if (Reason == MQRC_NO_MSG_AVAILABLE)
   /* Proceed to normal processing */
   ...

else
   /* Group was terminated prematurely */
   Set GroupId, MsgSeqNumber in MQMD to values from Status message
   PMO.Options = MQPMO_SYNCPOINT
   MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP

   /* Now normal processing is resumed.
      Assume this is not the last message */
   PMO.Options = MQPMO_LOGICAL_ORDER | MQPMO_SYNCPOINT
   MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
   MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP
   StatusInfo = GroupId,MsgSeqNumber from MQMD
   MQPUT (StatusInfo to STATUS queue) PMO.Options = MQPMO_SYNCPOINT
   MQCMIT
```

Alma uygulamasından, tüm grup gelmeden önce bir gruptaki iletileri işlemeye başlamak isteyebilirsiniz. Bu, grup içindeki iletilerde yanıt sürelerini iyileştirir ve ayrıca, tüm grup için depolamanın gerekli olmadığı anlamına gelir. Avantajların farkına varmak için, her ileti grubu için birkaç iş birimi kullanın. Kurtarma nedenlerinden dolayı, her bir iletiyi bir iş birimi içinde almanız gerekir.

Buna karşılık gelen uygulamada olduğu gibi, durum bilgilerinin her bir iş birimi kesinleştirildiğinde otomatik olarak bir yere kaydedilmesini gerektirir. Yine, bu bilgileri kaydetmenin en basit yeri STATUS kuyruğunda yer alıyor. Tam bir grup başarıyla işlendiyse, STATUS kuyruğu boş olur.

Not: Ara düzey iş birimleri için, her bir MQPUT işleminin bir ileti bölümü olduğunu (MQMF_SEGMENT işaretini ayarlayarak), her iş birimi için eksiksiz bir yeni ileti koymak yerine, STATUS kuyruğundan MQGET çağrılarını önleyebilirsiniz. Son iş biriminde, MQMF_LAST_SEGMENT değerini belirten durum kuyruğuna son bir kesim yerleştirilir ve MQGMO_COMPLETE_MSG belirtilerek, durum bilgileri bir MQGET ile temizlenir.

Yeniden başlatma işlemi sırasında, olası bir durum iletisi almak için tek bir MQGET kullanmak yerine, son kesime ulaşıncaya kadar (başka bir kesim döndürülünceye kadar) MQGMO_LOGICAL_ORDER ile durum kuyruğuna göz atın. Yeniden başlatmadan sonra ilk iş biriminde, durum bölümü yerleştirilirken açık olarak da göreli konum belirtin.

Aşağıdaki örnekte, yalnızca bir grup içindeki iletileri göz önünde bulundurarak, uygulamanın arabelleğinin tüm iletiyi tutmak için her zaman yeterince büyük olduğunu varsayarak, iletinin bölümlenip kesilmediğini dikkate alın. Bu nedenle, her MQGET üzerinde MQGMO_COMPLE_MSG belirtildi. Kesimlere ayırma ilişkisi varsa, aynı ilkeler geçerli olur (bu durumda, StatusInfo , *Offset* ' u içermelidir).

Basitlik için, tek bir UOW içinde en fazla 4 ileti alındığını varsayıyoruz:

```
msgs = 0    /* Counts messages retrieved within UOW */
/* Should be no status message at this point */
```

```
/* Retrieve remaining messages in the group */
do while ( GroupStatus == MQGS_MSG_IN_GROUP )

   /* Process up to 4 messages in the group */
   GMO.Options = MQGMO_SYNCPOINT | MQGMO_WAIT
               | MQGMO_LOGICAL_ORDER
   do while ( (GroupStatus == MQGS_MSG_IN_GROUP) && (msgs < 4) )
      MQGET
      msgs = msgs + 1
      /* Process this message */
      ...
   /* end while

   /* Have retrieved last message or 4 messages  */
   /* Update status message if not last in group */
   MQGET (from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
   if ( GroupStatus == MQGS_MSG_IN_GROUP )
      StatusInfo = GroupId,MsgSeqNumber from MQMD
      MQPUT (StatusInfo to STATUS queue) PMO.Options = MQPMO_SYNCPOINT
   MQCMIT
   msgs = 0
/* end while

if ( msgs > 0 )
   /* Come here if there was only 1 message in the group */
   MQCMIT
```

Tüm iş birimleri kesinleştirildiyse, tüm grup başarıyla alınır ve STATUS kuyruğu boş olur. Aksi takdirde, grubun durum bilgisiyle belirtilen noktada sürdürülmesi gerekir. MQGMO_LOGICAL_ORDER ilk alma işlemi için kullanılamıyor, ancak bundan sonra olabilir.

Yeniden başlatma işlemi şu şekilde görünür:

```
MQGET (from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
if (Reason == MQRC_NO_MSG_AVAILABLE)
   /* Proceed to normal processing */
   ...

else
   /* Group was terminated prematurely */
   /* The next message on the group must be retrieved by matching
      the sequence number and group id with those retrieved from the
      status information. */
   GMO.Options = MQGMO_COMPLETE_MSG | MQGMO_SYNCPOINT | MQGMO_WAIT
   MQGET GMO.MatchOptions = MQMO_MATCH_GROUP_ID | MQMO_MATCH_MSG_SEQ_NUMBER,
         MQMD.GroupId      = value from Status message,
         MQMD.MsgSeqNumber = value from Status message plus 1
   msgs = 1
   /* Process this message */
   ...

   /* Now normal processing is resumed */
   /* Retrieve remaining messages in the group */
   do while ( GroupStatus == MQGS_MSG_IN_GROUP )

      /* Process up to 4 messages in the group */
      GMO.Options = MQGMO_COMPLETE_MSG | MQGMO_SYNCPOINT | MQGMO_WAIT
                  | MQGMO_LOGICAL_ORDER
      do while ( (GroupStatus == MQGS_MSG_IN_GROUP) && (msgs < 4) )
         MQGET
         msgs = msgs + 1
         /* Process this message */
         ...

      /* Have retrieved last message or 4 messages  */
      /* Update status message if not last in group */
      MQGET (from STATUS queue) GMO.Options = MQGMO_SYNCPOINT
      if ( GroupStatus == MQGS_MSG_IN_GROUP )
         StatusInfo = GroupId,MsgSeqNumber from MQMD
         MQPUT (StatusInfo to STATUS queue) PMO.Options = MQPMO_SYNCPOINT
      MQCMIT
      msgs = 0
```

Belirli bir iletiyi alma

Bir kuyruktan belirli bir iletiyi almak için bir dizi yol vardır. Bunlar şunlardır: MsgId ve CorrelIdüzerinde seçim yapmak, GroupId, MsgSeqNumber and Offset seçeneğinden ve MsgToken' da seçilip seçilmediğiniz. Ayrıca, kuyruğu açtığınızda bir seçim dizgisi de kullanabilirsiniz.

Kuyruktan belirli bir ileti almak için, MQMD yapısının *MsgId* ve *CorrelId* alanlarını kullanın. Ancak, uygulamalar bu alanları belirtik olarak ayarlayabilirler, böylece belirttiğiniz değerler benzersiz bir iletiyi tanımlayamayabilir. Çizelge 38 sayfa 246 , bu alanların olası ayarları için hangi iletinin alındığını gösterir. MQGET çağrısının *GetMsgOpts* değiştirgesinde MQGMO_MSG_UNDER_CURSOR değerini belirlerseniz, bu alanlar girişte yoksayılır.

Çizelge 38. İleti ve ilinti tanıtıcılarının kullanılması

Almak için ...	***MsgId***	***CorrelId***
Kuyrukta ilk ileti	MQMI_NONE	MQCI_NONE
*MsgId*ile eşleşen ilk ileti	Sıfır Olmayan	MQCI_NONE
*CorrelId*ile eşleşen ilk ileti	MQMI_NONE	Sıfır Olmayan
Hem *MsgId* hem de *CorrelId*ile eşleşen ilk ileti	Sıfır Olmayan	Sıfır Olmayan

Her durumda *ilk* , seçim ölçütlerini karşılayan ilk ileti anlamına gelir (MQGMO_BROWSE_NEXT belirtildiyse, bu, seçim ölçütlerine uygun sırayla *sonraki* iletisi anlamına gelir).

Geri dönüşte, MQGET çağrısı *MsgId* ve *CorrelId* alanlarını, döndürülen iletinin ileti ve ilinti tanıtıcılarına ayarlar.

MQMD yapısının *Version* alanını 2 olarak ayarladıysanız, *GroupId*, *MsgSeqNumber*ve *Offset* alanlarını kullanabilirsiniz. Çizelge 39 sayfa 246 Bu alanların olası ayarları için hangi iletinin alındığını gösterir.

Çizelge 39. Grup tanıtıcısını kullanma

Almak için ...	**Eşleştirme seçenekleri**
Kuyrukta ilk ileti	MQMO_NONE
*MsgId*ile eşleşen ilk ileti	MQMO_MATCH_MSG_ID
*CorrelId*ile eşleşen ilk ileti	MQMO_MATCH_COREL_ID
*GroupId*ile eşleşen ilk ileti	MQMO_MATCH_GROUP_ID
*MsgSeqNumber*ile eşleşen ilk ileti	MQMO_MATCH_MSG_SEQ_NUMBER
*MsgToken*ile eşleşen ilk ileti	MQMO_MATCH_MSG_TOKEN
*Offset*ile eşleşen ilk ileti	MQMO_MATCH_OFFSET

Notlar:

1. MQMO_MATCH_XXX, MQMD yapısındaki *XXX* alanının, eşleştirilecek değere ayarlananını belirtir.
2. MQMO işaretleri birleşimde kullanılabilir. For example, MQMO_MATCH_GROUP_ID, MQMO_MATCH_MSG_SEQ_NUMBER, and MQMO_MATCH_OFFSET can be used together to give the segment identified by the *GroupId*, *MsgSeqNumber*, and *Offset* fields.
3. MQGMO_LOGICAL_ORDER belirtilirse, almaya çalıştığınız ileti etkilenir; bu seçenek, kuyruk tanıtıcısı için denetlenen durum bilgilerine bağlıdır. Bu konuda bilgi almak için bkz. “Mantıksal ve fiziksel sıralama” sayfa 235 ve Seçenekler.

MQGET çağrısı genellikle bir kuyruktan ilk iletiyi alır. MQGET çağrısını kullandığınızda belirli bir ileti belirtirseniz, kuyruk yöneticisi iletiyi buluncaya kadar kuyruğun aranması gerekir. Bu, uygulamanızın performansını etkileyebilir.

If you are using Version 2 or later of the MQGMO structure and do not specify the MQMO_MATCH_MSG_ID or MQMO_MATCH_CORREL_ID flags, you do not need to reset the *MsgId* or *CorrelId* fields between MQGETs.

MQGMO yapısındaki MsgToken değerini ve MatchOption MQMO_MATCH_MSG_TOKEN değerini belirterek bir kuyruktan belirli bir ileti alabilirsiniz. MsgToken , MQPUT çağrısıyla bu iletiyi kuyruğa ilk olarak koyan ya da önceki MQGET işlemleri tarafından döndürülür ve kuyruk yöneticisi yeniden başlatılmadıkça, sabit kalır.

Kuyrukta yalnızca bir ileti alt kümesiyle ilgileniyorsanız, MQOL ya da MQSUB çağrısıyla bir seçim dizgisi kullanarak işlenmesini istediğiniz iletileri belirleyebilirsiniz. Sonra MQGET, o seçim dizesini karşılayan bir sonraki iletiyi alır. Seçim dizgileri hakkında daha fazla bilgi için bkz. “Seçiciler” sayfa 21.

Kalıcı olmayan iletilerin performansını artırma

İstemci bir sunucudan ileti gerektirdiğinde, sunucuya bir istek gönderir. Tükettiği iletilerin her biri için ayrı bir istek gönderir. Bir istemcinin, bu istek iletilerini göndermek zorunda kalmaktan kaçınarak, kalıcı olmayan iletileri tüketen bir istemcinin performansını artırmak için, bir istemci *devamını oku*' yı kullanacak şekilde yapılandırılabilir. Önden okuma, bir uygulamanın istekte bulunmadan istemciye gönderilmesine izin verir.

When read ahead is enabled, messages are sent to a memory buffer on the client called the *okuma yazma arabelleği*. İstemci, önceden okuma etkinleştirilmiş olarak açık olduğu her kuyruk için okuma arabelleğiyle ilgili bir okuma arabelleği'yecektir. İleriye okuma arabelleğindeki iletiler kalıcı olarak saklanmaz. İstemci düzenli olarak sunucuyu, tükettiği veri miktarına ilişkin bilgilerle güncelleştirir.

MQOPEN ile MQO_READ_AHEAD ' i çağırdığınızda, WebSphere MQ istemcisi yalnızca belirli koşullar karşılandığında önden okuma seçeneğini etkinleştirir. Bu koşullar şunları içerir:

- İstemci ve uzak kuyruk yöneticisi, WebSphere MQ Sürüm 7 ya da sonraki bir sürümde olmalıdır.
- İstemci uygulaması, iş parçacıklı WebSphere MQ MQI istemci kitaplıklarına göre derlenmeli ve bağlantılandırılmalıdır.
- İstemci kanalının TCP/IP protokolünü kullanması gerekir
- Kanal, hem istemci hem de sunucu kanalı tanımlamalarında sıfır dışında bir SharingConversations (SHARECNV) ayarına sahip olmalıdır.

İleriyi okumanın kullanılması, bir istemci uygulamasından kalıcı olmayan iletiler tüketirken performansı yükseltebilirler. Bu performans iyileştirmesi, hem MQI hem de JMS uygulamaları için kullanılabilir. MQGET ya da zamanuyumsuz tüketimi kullanan istemci uygulamaları, kalıcı olmayan iletiler tüketildiğinde performans iyileştirmelerinden yararlanırlar.

İleriye okuma ile birlikte kullanım için tüm seçenekler desteklenmediği için, tüm istemci uygulama tasarımları okuma öncesinde kullanılması için uygun değildir ve önceden okuma etkinleştirildiğinde, MQGET çağrıları arasında tutarlı olması için bazı seçeneklerin kullanılması gerekir. Bir istemci, seçim ölçütlerini MQGET çağrıları arasında değiştirirse, ileriye okuma arabelleğindeki saklanmakta olan iletiler, istemcinin önünde okuma arabelleğindeki iplikçik olarak kalır.

Önceki seçim ölçütlerine sahip bir birikim arka günlüğü artık gerekmiyorsa, istemciden bu iletilerin istemciden otomatik olarak temizlenmesi için istemcide yapılandırılabilir bir temizleme aralığı ayarlanabilir. Temizleme aralığı, istemci tarafından belirlenen ileriye doğru ayarlama seçenekleri gruplarından biridir. Gereksinimlerinizi karşılamak için bu seçenekleri ayarlamak mümkündür.

Bir istemci uygulaması yeniden başlatılırsa, ileriye okuma arabelleğindeki iletiler kaybedilebilir. Tersi durumda, ileriye doğru okuma arabelleğiyle taşınan bir ileti, temeldeki kuyruktan silinebilir; bu, arabellekten kaldırılmamasına neden olmaz, bu nedenle okuma öncesinde kullanılan bir MQGET çağrısı, artık var olmayan bir iletiyi geri döndürebilir.

Önceden okuma, yalnızca istemci bağ tanımları için gerçekleştirilir. Diğer tüm bağ tanımlamalar için öznitelik yoksayılır.

İleriyi okumanın tetikleme konusunda hiçbir etkisi yoktur. İstemci tarafından ileriye doğru bir ileti okunduğunda tetikleme iletisi oluşturulmadı. Önceden okuma, etkinleştirildiğinde, muhasebe ve istatistik bilgileri oluşturmaz.

Yayınlama ile birlikte okuma yazma ileti alışverişi

Abone olunan bir uygulama, yayınların gönderildiği hedef kuyruğunu belirtiyorsa, varsayılan okuma değeri olarak, belirlenen kuyruğun DEFREADA değeri kullanılır.

WebSphere MQ ' un yayınların gönderildiği hedefi yönettiği bir uygulama isteği olduğunda, yönetilen bir kuyruk, önceden tanımlanmış bir model kuyruğuna dayalı olarak dinamik bir kuyruk olarak yaratılır. Bu, varsayılan okuma değeri değeri olarak kullanılan model kuyruğunun DEFREADA değeridir. Varsayılan model kuyrukları SYSTEM.DURABLE.PUBLICATIONS.MODEL ya da SYSTEM.NONDURABLE.PUBLICATIONS.MODEL , bu ya da bir üst konu için bir model kuyruğu tanımlanmadıkça kullanılır.

İlgili kavramlar

"AIXüzerinde kalıcı olmayan iletiler için performans ayarlaması" sayfa 250
AIX V5.3 ya da sonraki bir sürümünü kullanıyorsanız, ayarlama parametresini, kalıcı olmayan iletiler için tam performans kullanacak şekilde ayarlamayı düşünün.

İlgili görevler

"Okuma öncesinde okuma ve devre dışı bırakma" sayfa 249
Varsayılan olarak okuma seçeneği geçersiz kılınmaktadır. Okuma öncesinde okuma işlemini kuyrukta ya da uygulama düzeyinde etkinleştirebilirsiniz.

İlgili başvurular

"MQGET seçenekleri ve okuma önerisi" sayfa 248
Önceden okuma etkinleştirildiğinde tüm MQGET seçenekleri desteklenmez; MQGET çağrıları arasında tutarlı olması için bazı seçenekler gereklidir.

MQGET seçenekleri ve okuma önerisi

Önceden okuma etkinleştirildiğinde tüm MQGET seçenekleri desteklenmez; MQGET çağrıları arasında tutarlı olması için bazı seçenekler gereklidir.

MQOPEN ile MQO_READ_AHEAD ' i çağırdığınızda, WebSphere MQ istemcisi yalnızca belirli koşullar karşılandığında önden okuma seçeneğini etkinleştirir. Bu koşullar şunları içerir:

- İstemci ve uzak kuyruk yöneticisi, WebSphere MQ Sürüm 7 ya da sonraki bir sürümde olmalıdır.
- İstemci uygulaması, iş parçacıklı WebSphere MQ MQI istemci kitaplıklarına göre derlenmeli ve bağlantılandırılmalıdır.
- İstemci kanalının TCP/IP protokolünü kullanması gerekir
- Kanal, hem istemci hem de sunucu kanalı tanımlamalarında sıfır dışında bir SharingConversations (SHARECNV) ayarına sahip olmalıdır.

Aşağıdaki çizelge, ileriye okuma ve MQGET çağrıları arasında değiştirilip değiştirilmedikleri için hangi seçeneklerin desteklendiğini gösterir.

Çizelge 40. MQGET seçenekleri ve okuma önerisi

	Önceden okuma etkinleştirildiğinde izin verilir ve MQGET çağrıları arasında değiştirilebilir[5]	**Okuma önden okuma etkinleştirildiğinde izin verilir, ancak MQGET çağrıları arasında değiştirilemez[1]**	**Önceden okuma etkinleştirildiğinde izin verilmeyen MQGET Seçenekleri geçerli kılındığında[2]**
MQGET MQMD değerleri	MsgTnt[3] Koreli Tanıtıcı[3]	Kodlama CodedCharSetId	
MQGET MQGMO Seçenekleri	• MQGMO_NO_BEKLEME • MQGMO_BROWSE_MESSAGE _UNDER_CURSOR • MQGMO_BROWSE_FIRST • MQGMO_BROWSE_NEXT • MQGMO_FAIL_IF_QUIESCING	• MQGMO_SYNCPOINT_IF _PERSISTENSE • MQGMO_NO_SYNCPOINT • MQGMO_ACCEPT_TICHTED _MSG • MQGMO_CONVERT	• MQGMO_SET_SIGNAL • MQGMO_SYNCPOINT • MQGMO_MARK_SKIP _BACKUT • MQGMO_MSG_UNDER _CURSOR[4] • MQGMO_LOCK • MQGMO_UNLOCK • MQGMO_LOGICAL_ORDER • MQGMO_COMPLE_MSG • MQGMO_ALL_MSGS_AVALABILIR • MQGMO_ALL_SEGMENTS_ KULLANıLABILIR

Notlar:

1. Bu seçenekler MQGET çağrıları arasında değiştirilirse, bir MQRC_OPTIONS_CHANGED neden kodu döndürülür.
2. Bu seçenekler ilk MQGET çağrısında belirtilirse, önceden okuma geçersiz kılınır. İzleyen bir MQGET çağrısında bu seçenekler belirtilirse, MQRC_OPTIONS_ERROR neden kodunda bir neden kodu döndürülür.
3. Bir istemci uygulaması, MQGET çağrıları arasında MsgId ve CorrelId değerlerini değiştirirse, önceki değerlere sahip iletiler istemciye önceden gönderilmiş olabilir ve tüketilinceye kadar (ya da otomatik olarak temizlenen) istemcide önceden okuma arabelleğindeki iletiler kalır.
4. MQGMO_MSG_UNDER_CURSOR önden okuma ile olanaklı değil. Kuyruk açılırken, hem MQOO_BROWSE hem de MQOO_INPUT_SHARED ya da MQOO_INPUT_EXCLUSIVE seçeneklerinden biri belirtildiğinde, okuma öncesinde okuma geçersiz kılınır.
5. İleriye okuma etkinleştirildiğinde, iletilerin bir kuyruktan mı göz atılacağını, yoksa kuyruktan mı atılacağını ilk MQGET belirler. İstemci uygulaması değiştirilen seçeneklerle MQGET işlevini kullanıyorsa (örneğin, bir ilk alma işlemini izlemeye ya da ilk göz atma işlemini izlemeye çalışmak gibi), bir MQRC_OPTIONS_CHANGED neden kodu döndürülür.

Bir istemci, seçim ölçütlerini MQGET çağrıları arasında değiştirirse, ilk seçim ölçütleriyle eşleşen okuma öncesinde saklanan iletiler istemci uygulaması tarafından tüketilmez ve istemci okuma değeri arabelleğindeki iplikçik olarak kalmaya devam eder. İstemcinin önceden okuma arabelleğindeki birçok hata içeren ileti içerdiği durumlarda, önden okuma ile ilişkili yararlar kaybedilir ve her ileti tüketilen her ileti için sunucuya ayrı bir istek gerekir. Önden okuma işleminin etkin bir şekilde kullanılıp kullanılmadığını belirlemek için, READA bağlantı durumu parametresini kullanabilirsiniz.

İlk MQGET çağrısında belirtilen uyumsuz seçenekler nedeniyle, bir uygulama tarafından istendiğinde okuma engelleyici olabilir. Bu durumda, bağlantı durumu engellendikçe öndeki okunurları gösterir.

MQGET ile ilgili bu kısıtlamalardan ötürü, ileride bir istemci uygulaması tasarımının okunmaya uygun olmadığını, MQOO_READ_AHEAD_NO adlı MQOO_READE ' yi belirtin. Diğer bir seçenek olarak, açılmakta olan kuyruğun varsayılan okuma tamamlama değeri NO ya da DISABLE olarak değiştiriliyor.

Okuma öncesinde okuma ve devre dışı bırakma

Varsayılan olarak okuma seçeneği geçersiz kılınmaktadır. Okuma öncesinde okuma işlemini kuyrukta ya da uygulama düzeyinde etkinleştirebilirsiniz.

Bu görev hakkında

MQOPEN ile MQO_READ_AHEAD ' i çağırdığınızda, WebSphere MQ istemcisi yalnızca belirli koşullar karşılandığında önden okuma seçeneğini etkinleştirir. Bu koşullar şunları içerir:

- İstemci ve uzak kuyruk yöneticisi, WebSphere MQ Sürüm 7 ya da sonraki bir sürümde olmalıdır.
- İstemci uygulaması, iş parçacıklı WebSphere MQ MQI istemci kitaplıklarına göre derlenmeli ve bağlantılandırılmalıdır.
- İstemci kanalının TCP/IP protokolünü kullanması gerekir
- Kanal, hem istemci hem de sunucu kanalı tanımlamalarında sıfır dışında bir SharingConversations (SHARECNV) ayarına sahip olmalıdır.

İleriyi okumayı etkinleştirmek için:

- Okuma öncesinde kuyruk düzeyinde okuma yapılandırmak için, kuyruk özniteliğini, DEFREADA ' yı YES değerine ayarlayın.
- Uygulama düzeyinde okuma öbeklerini yapılandırmak için:
  - MQOPEN işlev çağrısındaki MQOO_READ_AHEAD seçeneğini mümkün olan her yerde kullanmak için okuma seçeneğini kullanın. DEFREADA kuyruk özniteliği DISABLE olarak ayarlandıysa, istemci uygulamasının okuma öncesinde okuma kullanması mümkün değildir.
  - Bir kuyruğun üzerinde okuma yazma özelliği etkinleştirildiğinde, önceden okunmayı kullanmak için, MQOPER işlev çağrısında MQOO_READ_AHEAD_AS_Q_DEF seçeneğini kullanın.

İleride bir istemci uygulaması tasarımı okumak için uygun değilse, bu tasarımı devre dışı bırakabilirsiniz:

- Kuyruk özniteliğini ayarlayarak, bir istemci uygulaması tarafından istenmedikçe, önceden okunmasını istemiyorsanız, DEFREOKA 'yı HAYIR olarak ayarlayarak, önceden okunmanın bir istemci uygulaması tarafından gerektirip okunmamasından bağımsız olarak okunmasını istemiyorsanız, DEVRE Dışı olarak kullanılmasını istemiyorsanız, DEFREADA' da (Hayır) kullanılmasını istemiyorsanız, DEFREADA ' yı (NO) belirleyerek, önceden okuma işlemi yapmanız gerekmez.
- MQOP işlev çağrısındaki MQOO_NO_READ_AHEAD seçeneğini kullanarak uygulama düzeyinde.

İki MQCLOSE seçeneği, kuyruk kapatılırsa, okuma öncesinde okuma arabelleğinde saklanan iletilere ne olacağını yapılandırmanızı sağlar.

- Okuma öbekleri arabelleğindeki iletileri atmak için MQCO_IMMEDIATE deyimini kullanın.
- İleriye okuma arabelleğindeki iletilerin, kuyruk kapatılmadan önce uygulama tarafından tüketildiğinden emin olmak için MQCO_QUIESCE seçeneğini kullanın. MQCO_QUIESCE ile MQCLOSE komutu verildiğinde ve ileriye doğru okuma arabelleğinde kalan iletiler varsa, MQRC_READ_AHEAD_MSGS, MQCC_UYARI ile döner.

AIXüzerinde kalıcı olmayan iletiler için performans ayarlaması

AIX V5.3 ya da sonraki bir sürümünü kullanıyorsanız, ayarlama parametresini, kalıcı olmayan iletiler için tam performans kullanacak şekilde ayarlamayı düşünün.

Ayarlama parametresini hemen yürürlüğe girebilmesi için ayarlamak üzere kök kullanıcı olarak şu komutu verin:

```
/usr/sbin/ioo -o j2_nPagesPerWriteBehindCluster=0
```

Eniyileme değiştirgesini hemen yürürlüğe gireceği ve yeniden başlatma işlemi üzerinde devam edecek şekilde ayarlamak için, kök kullanıcı olarak aşağıdaki komutu verin:

```
/usr/sbin/ioo -p -o j2_nPagesPerWriteBehindCluster=0
```

Olağan durumda, kalıcı olmayan iletiler yalnızca bellekte tutulur, ancak AIX , kalıcı olmayan iletileri diske yazılacak şekilde zamanlayabileceği durumlar da vardır. Diske yazılması planlanan iletiler, disk yazma işlemi tamamlanıncaya kadar MQGET için kullanılamaz. Önerilen ayarlama komutu bu eşik değerini gösterir; 16 kilobayt veri kuyruğa alındığında iletilerin diske yazılacağı zamanlamak yerine, diske yazma işlemi yalnızca makineden gerçek saklama alanı tam olarak kapandığında gerçekleşir. Bu, genel bir değişiktir ve diğer yazılım bileşenlerini etkileyebilir.

AIXişletim sistemi, çok iş parçacıklı uygulamalar kullanırken ve özellikle birden çok işlemcili makinelerde çalışırken, uygulama başlatılmadan önce ortamda AIXTHREAD_SCOPE=S ayarını `.profile` mqm tanıtıcısında ya da AIXTHREAD_SCOPE=S ayarlamasını, daha iyi performans ve daha sağlam zamanlama için önemle öneririz. Örneğin:

```
export AIXTHREAD_SCOPE=S
```

AIXTHREAD_SCOPE=S ayarı, varsayılan özniteliklerle yaratılan kullanıcı iş parçacıklarının sistem çapında çekişme kapsamına konacağı anlamına gelir. Sistem çapında çekişme kapsamı ile bir kullanıcı iş parçacığı yaratılırsa, bu iş parçacığı bir çekirdek iş parçacığa bağlanır ve bu iş parçacığın çekirdeğe göre zamanlanır. Temeldeki çekirdek iş parçacığı, başka bir kullanıcı iş parçacığıyla paylaşılmaz.

Dosya tanımlayıcıları

Aracı işlemi gibi çok iş parçacıklı bir işlemi çalıştırırken, dosya açıklayıcıları için yumuşak sınıra ulaşabilirsiniz. Bu sınır, size IBM WebSphere MQ neden kodu `MQRC_UNEXPECTED_ERROR (2195)` ve yeterli dosya tanımlayıcısı varsa, IBM WebSphere MQ FFST™ dosyası verir.

Bu sorunu önlemek için, dosya tanımlayıcılarının sayısı için süreç sınırını artırabilirsiniz. To do so, alter the `nofiles` attribute in `/etc/security/limits` to 10,000 for the mqm user ID or in the default stanza.

Sistem Kaynağı Sınırları

Bir komut isteminde aşağıdaki komutları kullanarak, veri bölümü ve yığın bölümü için sistem kaynağı sınırını sınırsız olarak ayarlayın:

```
ulimit -d unlimited
ulimit -s unlimited
```

4 MB ' den büyük iletilerin işlenmesi

İletiler, uygulama, kuyruk ya da kuyruk yöneticisi için çok büyük olabilir. Ortama bağlı olarak, WebSphere MQ , 4 MB ' den uzun iletilerle ilgilenmenin çeşitli yollarını sağlar.

MaxMsgLength özniteliğini, V6 ya da daha sonraki bir yayın düzeyinde tüm WebSphere MQ sistemlerinde 100 MB ' ye yükseltebilirsiniz. Kuyruğu kullanan iletilerin büyüklüğünü yansıtmak için bu değeri ayarlayın. On WebSphere MQ systems other than WebSphere MQ for z/OS, you can also:

1. Kesimlere ayrılmış iletileri kullanın. (İletiler, uygulama ya da kuyruk yöneticisi tarafından kesimlere ayrılabilir.)
2. Başvuru iletilerini kullanın.

Bu yaklaşımların her biri bu bölümün geri kalan kısmında açıklanmıştır.

İleti uzunluğu üst sınırını artırma

MaxMsgLength kuyruk yöneticisi özniteliği, bir kuyruk yöneticisi tarafından işlenebilecek bir iletinin uzunluk üst sınırını tanımlar. Benzer şekilde, *MaxMsgLength* kuyruk özniteliği, bir kuyruk tarafından işlenebilecek bir iletinin uzunluk üst sınışıdır. Desteklenen *varsayılan* ileti uzunluğu üst sınırı, çalışmakta olduğunuz ortama bağlıdır.

Büyük iletileri ele aldıysanız, bu öznitelikleri bağımsız olarak değiştirebilirsiniz. Kuyruk yöneticisi özniteliği değerini 32768 bayt-100 MB aralığında ayarlayabilirsiniz; kuyruk özniteliği değerini 0-100 MB aralığında ayarlayabilirsiniz.

MaxMsgLength özniteliklerinin birini ya da her ikisini değiştirdikten sonra, değişikliklerin yürürlüğe girdiğinden emin olmak için uygulamalarınızı ve kanallarınızı yeniden başlatın.

Bu değişiklikler yapıldığında, ileti uzunluğu hem kuyruktan, hem de kuyruk yöneticisi *MaxMsgLength* özniteliklerinden küçük ya da ona eşit olmalıdır. Ancak, var olan iletiler özniteliklerden daha uzun olabilir.

İleti kuyruk için çok büyükse, MQRC_MSG_TOO_BIG_FOR_Q döndürülür. Benzer şekilde, ileti kuyruk yöneticisi için çok büyükse, MQRC_MSG_TOO_BIG_FOR_Q_MGR iletisi döndürülür.

Büyük iletilerin ele alma yöntemi kolay ve uygundur. Ancak, aşağıdaki etkenleri kullanmadan önce aşağıdaki etkenleri göz önünde bulundurun:

- Kuyruk yöneticileri arasındaki benzersizlik azaltılır. İleti verilerinin büyüklük üst sınırı, iletinin konacağı her kuyruk (iletim kuyrukları da içinde olmak üzere) için *MaxMsgLength* tarafından belirlenir. Bu değer genellikle, kuyruk yöneticisinin *MaxMsgLength*, özellikle de iletim kuyrukları için varsayılan olarak varsayılan olarak belirlenir. Bu, uzak bir kuyruk yöneticisine seyahat etmek olduğunda iletinin çok büyük olup olmadığını tahmin etmeyi zorlaştırır.
- Sistem kaynaklarının kullanımı artırılır. Örneğin, uygulamaların daha büyük arabelleklere ve bazı platformlarda, paylaşılan saklama alanı kullanımı artırılmış olabilir. Kuyruk depolaması yalnızca daha büyük iletiler için gerekliyse etkilenmelidir.
- Kanal toplu işi etkileniyor. Büyük bir ileti, toplu iş sayısına doğru yalnızca bir ileti olarak sayılır, ancak iletilmeye daha uzun gereksinim duyar, böylece diğer iletiler için yanıt sürelerini artırır.

İleti bölümlemesi

Bölüm iletileri hakkında bilgi edinmek için bu bilgileri kullanın.

Not: Not supported in IBM WebSphere MQ for z/OS or by applications using IBM WebSphere MQ classes for JMS.

"İleti uzunluğu üst sınırını artırma" sayfa 251 konusunda açıklandığı gibi ileti uzunluğu üst sınırının artırılması bazı olumsuz etkilerine yol açıyor. Ayrıca, ileti kuyruğun ya da kuyruk yöneticisi için çok büyük olmasına neden olabilir. Bu durumda, bir iletiyi bölümleyebilirsiniz. Bölümlerle ilgili bilgi için bkz. "İleti grupları" sayfa 34.

Sonraki bölümler, bölümlenmiş iletiler için ortak kullanımlara bakın. Koyma ve geçici olarak alma işlemi için, *her zaman* MQPUT ya da MQGET çağrılarının bir iş birimi içinde çalışması olduğu varsayılır. Bu tekniği her zaman, ağda eksik grupların mevcut olma olasılığını azaltmak için kullanmayı düşünün. Kuyruk yöneticisi tarafından tek aşamalı kesinleştirme kabul edilir, ancak diğer koordinasyon teknikleri eşit olarak geçerlidir.

Ayrıca, uygulama alma işlemlerinde, aynı kuyruğu birden çok sunucu işliyorsa, her bir sunucu benzer kodu yürütür; böylece, bir sunucu, orada olmayı beklediği bir ileti ya da kesimi bulamazsa (daha önce MQGMO_ALL_MSGS_AVAILD ya da MQGMO_ALL_SEGMENTS_AVASILABILIR belirtmiş olduğu için) hiçbir zaman başarısız olur.

İş birimlerine yayılan bölümlü bir ileti alınıyor ve alınıyor

"İş birimlerine yayılan bir grubu koymak ve almak" sayfa 243' e benzer bir şekilde bir çalışma birimine yayılan bölümlü bir ileti yerleştirebilir ve alabilirsiniz.

Ancak, bölümlenmiş iletileri genel bir iş birimine yerleştiremez ya da bu iletileri bölümlere ayırtamazsınız.

Kuyruk yöneticisine göre bölümlendirme ve yeniden birleştirme
Bu, bir uygulamanın başka bir uygulama tarafından alınması gereken bir iletiyi yerleştirdiği en basit senaryodur. İleti büyük olabilir: put ya da alma uygulaması tek bir arabelleğiyle başa çıkmak için çok büyük değil, ancak kuyruk yöneticisi için çok büyük ya da iletinin konacağı bir kuyruk için çok büyük.

Bu uygulamalar için gerekli olan tek değişiklikler, uygulamanın kuyruk yöneticisine, gerekli durumlarda segmentasyon gerçekleştirmesi için yetki vermesi amacıyla gerçekleştirilir:

```
PMO.Options = (existing options)
MD.MsgFlags = MQMF_SEGMENTATION_ALLOWED
memcpy(MD.GroupId, MQGI_NONE, MQ_GROUP_ID_LENGTH)
MQPUT
```

ve uygulama alma işlemi için, kuyruk yöneticisine, bölümlenmiş bir ileti varsa, iletiyi yeniden bir araya getirmesini istemesi için:

```
GMO.Options = MQGMO_COMPLETE_MSG | (existing options)
MQGET
```

Bu en basit senaryoda, uygulama, MQPUT çağrısından önce GroupId alanını MQGI_NONE olarak sıfırlamalıdır, böylece kuyruk yöneticisi her ileti için benzersiz bir grup tanıtıcısı oluşturabilirler. Bu yapılmıyorsa, ilgisiz iletiler aynı grup tanıtıcısına sahip olabilir ve bu da daha sonra yanlış işleme yol açabilir.

Uygulama arabelleği, yeniden birleştirilen iletiyi içerecek kadar büyük olmalıdır (MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini eklemediyseniz).

Bir kuyruğun MAXMSGLEN özniteliği, ileti bölümlemesi için değiştirilecek şekilde değiştirilecekse, şunları dikkate alın:

- Yerel kuyruklarda desteklenen ileti kesimi alt sınırı 16 byte 'tır.
- İletim kuyruğu için, MAXMSGLEN, üstbilgiler için gereken alanı da içermelidir. İletim kuyruğuna konabilecek herhangi bir ileti kesiminde, beklenen kullanıcı verisi uzunluğu üst sınırından en az 4000 bayt daha büyük bir değer kullanın.

Veri dönüştürme gerekliyse, uygulama alma işlemi MQGMO_CONVERT belirterek bunu yapmak zorunda kalabilirler. Veri dönüştürme çıkışı eksiksiz bir iletiyle sunulduğu için bu açık bir şekilde olmalıdır.
İleti bölümlendiyse, verileri gönderen kanalına dönüştürme girişiminde bulunmayın; verilerin biçimi veri dönüştürme çıkışıysa, eksik verilerde dönüştürme işlemini gerçekleştiremez.

Uygulama bölümlemesi
Uygulama kesimleme, kuyruk yöneticisi bölümlemesi yeterli değilse ya da uygulamalar belirli bölüm sınırları ile veri dönüştürme gerektirdiğinde kullanılır.

Uygulama bölümlemesi iki temel neden için kullanılır:

1. İleti, uygulamalar tarafından tek bir arabelleğde işlenmeyecek kadar büyük olduğundan, yalnızca kuyruk yöneticisi bölümlemesi yeterli değil.
2. Veri dönüştürme, gönderen kanalları tarafından gerçekleştirilmelidir ve biçim, bir kesimin tek bir kesimin mümkün olması için kesim sınırlarının nerede olacağını belirtmesi gerektiğini ifade eder.

Ancak, veri dönüştürme bir sorun değilse ya da alma uygulaması her zaman MQGMO_COMPLETE_MSG kullanırsa, kuyruk yöneticisi bölümlemesine izin verilirse, MQMF_SEGMENTATION_ALLOWEND belirtilerek de izin verilir. Örneğimizde, uygulama iletiyi dört kesime ayırır:

```
PMO.Options = MQPMO_LOGICAL_ORDER | MQPMO_SYNCPOINT

MQPUT MD.MsgFlags = MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_LAST_SEGMENT

MQCMIT
```

If you do not use MQPMO_LOGICAL_ORDER, the application must set the *Offset* and the length of each segment. Bu durumda, mantıksal durum otomatik olarak korunmaz.

Uygulama alma işlemi, yeniden birleştirilen bir iletiyi tutacak kadar büyük bir arabelleğe sahip olduğunu garanti edemez. Bu nedenle, kesimleri tek tek işlemeye hazırlanmalıdır.

Bölümlenmiş iletiler için, bu uygulama, mantıksal iletiyi oluşturan tüm
kesimler varsa, bir bölümü işlemeye başlamak istemiyor. Bu nedenle, ilk
bölüm için MQGMO_ALL_SEGMENTS_AVAILLLLLLLLLLLLLLLLLLLR MQGMO_LOGICAL_ORDER
öğesini belirtirseniz ve yürürlükteki bir mantıksal ileti varsa,
MQGMO_ALL_SEGMENTS_AVAILLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLR.

Mantıksal iletinin ilk bölümü alındıktan sonra, mantıksal iletinin kalan bölümlerinin sırayla alındığından emin olmak için MQGMO_LOGICAL_ORDER kullanın.

Farklı gruplar içindeki iletilere göz önünde bulunmuklanmaz. Bu tür iletiler gerçekleşirse, kuyrukta her iletinin ilk bölümünün gerçekleştirildiği sırayla işlenir.

```
GMO.Options = MQGMO_SYNCPOINT | MQGMO_LOGICAL_ORDER
            | MQGMO_ALL_SEGMENTS_AVAILABLE | MQGMO_WAIT
do while ( SegmentStatus == MQSS_SEGMENT )
   MQGET
   /* Process each remaining segment of the logical message */
   ...

MQCMIT
```

Mantıksal iletilerin uygulama bölümlemesi
İletiler, bir gruptaki mantıksal sırada sürdürülmelidir; bunların bazıları ya da tümü, uygulama bölümlemesi gerektirmeleri için bu kadar büyük olabilir.

Örneğimizde, dört mantıksal ileti içeren bir grup ortaya konabiliyor. Üçüncü iletinin tümü büyük, ancak uygulama koyma işlemi tarafından gerçekleştirilen bölümlemeye gerek duyarsınız:

```
PMO.Options = MQPMO_LOGICAL_ORDER | MQPMO_SYNCPOINT

MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP      | MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP      | MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP      | MQMF_LAST_SEGMENT

MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP      | MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP      | MQMF_LAST_SEGMENT
```

```
MQPUT MD.MsgFlags = MQMF_MSG_IN_GROUP

MQPUT MD.MsgFlags = MQMF_LAST_MSG_IN_GROUP | MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_LAST_MSG_IN_GROUP | MQMF_SEGMENT
MQPUT MD.MsgFlags = MQMF_LAST_MSG_IN_GROUP | MQMF_LAST_SEGMENT

MQCMIT
```

Uygulama alma uygulamasında, ilk MQGET ' de MQGMO_ALL_MSGS_AVAM değeri belirtildi. Bu, grubun tamamı kullanılabilir oluncaya kadar bir grubun hiçbir ileti ya da kesiminin alınmamasını sağlar. Bir grubun ilk fiziksel iletisi alındığında, grubun bölümlerinin ve iletilerinin sırayla alındığından emin olmak için MQGMO_LOGICAL_ORDER kullanılır:

```
GMO.Options = MQGMO_SYNCPOINT | MQGMO_LOGICAL_ORDER
            | MQGMO_ALL_MSGS_AVAILABLE | MQGMO_WAIT

do while ( (GroupStatus   != MQGS_LAST_MSG_IN_GROUP) ||
             (SegmentStatus != MQGS_LAST_SEGMENT) )
   MQGET
   /* Process a segment or complete logical message.  Use the GroupStatus
      and SegmentStatus information to see what has been returned */
   ...

MQCMIT
```

Not: MQGMO_LOGICAL_ORDER öğesini belirtirseniz ve yürürlükteki bir grup varsa, MQGMO_ALL_MSGS_AVAM yok sayılır.

Başvuru iletileri

Başvuru iletileri hakkında daha fazla bilgi edinmek için bu bilgileri kullanın.

Not: z/OSiçin WebSphere MQ ' da desteklenmez.

Bu yöntem, büyük bir nesnenin bir düğümden diğerine, kaynak ya da hedef düğümlerde WebSphere MQ kuyruklarında saklanmadan başka bir düğümden diğerine aktarılmasına olanak sağlar. Bu, örneğin, posta uygulamaları için başka bir formda (örneğin, başka bir formda) var olduğunda bu avantajdan yararlanmaktan çok daha iyi olur.

Bunu yapmak için, bir kanalın her iki ucunda da bir ileti çıkışı belirtiyorsunuz. Bunun nasıl yapacağına ilişkin bilgi için bkz. “Kanal ileti çıkış programları” sayfa 393.

WebSphere MQ , bir başvuru iletisi üstbilgisinin biçimini (MQRMH) tanımlar. Bu konuya ilişkin açıklamalar için MQRMH başlıklı konuya bakın. Bu, tanımlı bir biçim adıyla tanınır ve gerçek veriler tarafından izlenebilir.

Bir uygulama, büyük bir nesnenin aktarımı başlatmak için, aşağıdaki verileri izleyen herhangi bir veri olmadan, başvuru iletisi üstbilgisinden oluşan bir ileti yerleştirebilir. Bu ileti düğümü bıraktıkça, ileti çıkışı nesneyi uygun bir şekilde alır ve başvuru iletisine ekler. Daha sonra iletiyi almak üzere Message Channel Agent 'ı göndermek üzere gönderilen Message Channel Agent 'a (öncekinden daha büyük) iletiyi döndürür.

Alıcı MCA ' da başka bir ileti çıkışı yapılandırılıyor. When this message exit receives one of these messages, it creates the object using the object data that was appended and passes on the reference message *bu olmadan* it. Başvuru iletisi artık bir uygulama tarafından alınabilir ve bu uygulama, bu düğümde nesnenin (ya da en azından bu başvuru iletisi tarafından temsil edilen kısmı) oluşturulduğunu biliyor.

Bir ileti çıkışısının başvuru iletisine ekleyebileceği nesne verisi miktarı üst sınırı, kanal için kararlaştırılan ileti uzunluğu üst sınırı ile sınırlanır. Çıkış, iletileceği her ileti için MCA ' ya yalnızca tek bir ileti döndürebilir; bu nedenle, uygulama koyma işlemi bir nesnenin aktarılmasına neden olacak birkaç ileti koyabilir.
Her ileti, sonuna eklenecek nesnenin *mantıksal* uzunluğunu ve göreli konumunu tanımlamalıdır. Ancak, nesnenin toplam büyüklüğünü ya da kanal tarafından izin verilen büyüklük üst sınırını bilmenin mümkün olmadığı durumlarda, gönderme ileti çıkışını tasarlayın; böylece, koyma işlemi tek bir ileti koyar ve çıkış, aktarıldığı iletiye olabildiğince çok veri eklendiğinde, bir sonraki iletiyi iletim kuyruğuna yerleştirir.

Büyük iletilerle ilgili olarak bu yöntemi kullanmadan önce aşağıdaki noktaları göz önünde bulundurun:

- MCA ve ileti çıkışı bir WebSphere MQ kullanıcı kimliği altında çalıştırılıyor. İleti çıkışı (ve dolayısıyla, kullanıcı kimliği), gönderme uçta almak ya da alıcı uçta yaratmak için nesneye erişmesi gerekir; bu durum, yalnızca nesnenin evrensel olarak erişilebilir olduğu durumlarda uygulanabilir. Bu da bir güvenlik sorunu yaratıyor.
- Sonuna kadar yığın verileri eklenmiş olan başvuru iletisi, hedefine ulaşmadan önce birkaç kuyruk yöneticisi aracılığıyla seyahat etmek zorunda olursa, girişimsel düğümlerdeki WebSphere MQ kuyruklarında bulunanbulktoplu veri *:* kuyrukları vardır. Ancak, bu durumlarda özel destek ya da çıkışlar sağlanmaz.
- Yeniden yönlendirme ya da ölü harf kuyruğa alma işlemine izin veriliyorsa, ileti çıkışınızın tasarlanması zorlaşmanızı sağlar. Bu durumlarda, nesnenin bölümleri siparişten çıkabilirler.
- Bir başvuru iletisi hedefine ulaştığında, alıcı ileti çıkışı nesneyi yaratır. Ancak, bu, MCA ' nın iş birimi ile uyumlulaştırılmaz; dolayısıyla, toplu iş yedeklenirse, nesnenin bu aynı bölümünü içeren başka bir başvuru iletisi daha sonraki bir kümeye gelecek ve ileti çıkışı, nesnenin aynı bölümünü yeniden yaratmayı deneyebilir. Nesne örneğin, bir dizi veritabanı güncelleştirmesi ise, bu kabul edilemez olabilir. Bu durumda, ileti çıkışı, güncellemelerin uygulanmış olduğu bir günlüğü tutmalıdır; bu, bir WebSphere MQ kuyruğu kullanılmasını gerektirebilir.
- Nesne tipinin özelliklerine bağlı olarak, ileti çıkışlar ve uygulamaların, kullanım sayılarının korunması konusunda işbirliği yapması gerekebilir; böylece, nesne artık gerekmediği zaman silinebilir. Bir yönetim ortamı tanıtıcısı da gerekli olabilir; başvuru iletisi üstbilgisinde bu alan için bir alan sağlanır (bkz. MQRMH).
- Bir başvuru iletisi dağıtım listesi olarak konulursa, o düğümdeki her bir dağıtım listesi ya da her bir hedef için nesnenin yeniden alınması gerekir. Kullanım sayılarını korumanız gerekebilir. Ayrıca, bir düğümün listedeki bazı hedefler için son düğüm, ancak diğer kullanıcılar için bir ara düzey düğüm olabileceği olasılığını da göz önünde bulundurun.
- Toplu veriler genellikle dönüştürülmez. Bunun nedeni, dönüştürme *önce* ' un ileti çıkışa çağrıldığı yer almasıdır. Bu nedenle, kaynak gönderen kanalda dönüştürme isteğinde bulunulmamalıdır. Başvuru iletisi bir ara düğümden geçerse, yığın veriler istenirse ara düğümden gönderildiğinde dönüştürülür.
- Başvuru iletileri kesimlere ayrılamaz.

MQRMH ve MQMD yapılarının kullanılması

Başvuru iletisi üstbilgisindeki alanların ve ileti tanımlayıcısının bir açıklaması için MQRMH ve MQMD başlıklı konuya bakın.

MQMD yapısında, *Format* alanını MQFMT_REF_MSG_HEADER olarak ayarlayın. The MQHREF format, when requested on MQGET, is converted automatically by WebSphere MQ along with any bulk data that follows.

Aşağıda, MQRMH ' nin *DataLogicalOffset* ve *DataLogicalLength* alanlarının kullanımına ilişkin bir örnek yer almaktadır:

Bir uygulama koyma işlemi şu iletiyle bir başvuru iletisi gönderebilir:

- Fiziksel veri yok
- *DataLogicalLength* = 0 (bu ileti, tüm nesneyi gösterir)
- *DataLogicalOffset* = 0.

Nesnenin 70 000 bayt uzunluğunda olduğunu varsayarsak, ileti gönderme çıkışı kanal boyunca ilk 40 000 baytı aşağıdaki ileti içeren bir başvuru iletisinde gönderir:

- MQRMH ' nin ardından 40 000 bayt fiziksel veri
- *DataLogicalLength* = 40000
- *DataLogicalOffset* = 0 (nesnenin başlangıcından).

Daha sonra, aşağıdaki ileti içeren iletim kuyruğunda başka bir ileti yerleştirir:

- Fiziksel veri yok
- *DataLogicalLength* = 0 (nesnenin sonuna kadar). Burada 30 000 değerini belirtebilirsiniz.

- *DataLogicalOffset* = 40000 (bu noktadan başlayarak).

Bu ileti çıkışı gönderme iletisi çıkışı tarafından görüldüğünde, geri kalan 30 bin bayt veri sonuna eklenir ve bu alanlara aşağıdaki alanlar ayarlanır:

- MQRMH ' nin ardından 30 bin bayt fiziksel veri
- *DataLogicalLength* = 30000
- *DataLogicalOffset* = 40000 (bu noktadan başlayarak).

MQRMHF_SON işareti de ayarlıdır.

Başvuru iletileri kullanımı için sağlanan örnek programların bir açıklaması için bkz. “Dağıtılmış platformlar için örnek programlar” sayfa 92.

İleti bekleniyor

Bir programın kuyrukta bir ileti gelene kadar beklemesini istiyorsanız, MQGMO yapısının *Options* alanında MQGMO_WEKE seçeneğini belirtin.

Belirtmek için MQGMO yapısının *WaitInterval* alanını kullanın.MQGET çağrısının bir iletinin kuyruğa gelmesini beklemesini istediğiniz süre üst sınırı (milisaniye cinsinden).

İleti bu süre içinde gelmezse, MQGET çağrısı MQRC_NO_MSG_AVAILEABLE neden koduyla tamamlanır.

WaitInterval alanında, sabit MQWI_UNESSıNI değerini kullanarak sınırsız bekleme aralığı belirtebilirsiniz. Ancak, denetiminiz dışındaki olaylar programınızın uzun bir süre beklemesine neden olabilir, bu nedenle bu değişmezi dikkatli kullanın. IMS applications must not specify an unlimited wait interval because this would prevent the IMS system terminating. (IMS sona erdirildiğinde, tüm bağımlı bölgelerin sona ermesini gerektirir.) Bunun yerine, IMS uygulamaları sonlu bekleme aralığını belirtebilir; daha sonra, arama tamamlandıktan sonra bir ileti alınmadan arama tamamlanırsa, bekleme seçeneğiyle başka bir MQGET çağrısı yayınlayın.

Not: Bir iletiyi *kaldırmak* için aynı paylaşılan kuyruktan birden fazla program bekliyorsa, gelen bir ileti tarafından yalnızca bir program etkinleşir. Ancak, bir iletiye göz atmak için birden fazla program bekliyorsa, tüm programlar etkinleştirilebilir. Daha fazla bilgi için, MQGMO' daki MQGMO yapısının *Options* alanının açıklamasına bakın.

Kuyruğun durumu ya da bekleme süresi dolmadan önce kuyruk yöneticisi değişirse, aşağıdaki işlemler gerçekleşir:

- Kuyruk yöneticisi susturma durumuna girerse ve MQGMO_FAIL_IF_QUIESCING seçeneğini kullandıysanız, bekleme iptal edilir ve MQGET çağrısı MQRC_Q_MGR_QUIESCING neden koduyla tamamlanır. Bu seçenek olmadan, arama işlemi beklemeye devam eder.
- Kuyruk yöneticisi durdurulmaya zorlandıysa ya da iptal edildiyse, MQGET çağrısı MQRC_Q_MGR_STOPPING ya da MQRC_CONNECTION_BROKEN neden koduyla tamamlanır.
- İsteklerin artık engellenebilmesi için kuyruğun öznitelikleri (ya da kuyruk adı çözümleyicilerinin bulunduğu bir kuyruk) varsa, bekleme iptal edilir ve MQGET çağrısı, MQRC_GET_INHIBITED neden koduyla tamamlanır.
- FORCE seçeneğinin gerekli olduğu bir şekilde kuyruğun öznitelikleri (ya da kuyruk adı çözümleyicilerinin bulunduğu bir kuyruk) değişirse, bekleme işlemi iptal edilir ve MQGET çağrısı MQRC_OBJECT_CHANGED neden koduyla tamamlanır.

Bu işlemlerin gerçekleştirildiği durumlarla ilgili daha fazla bilgi için bkz. MQGMO.

Geri alma işlemi atlanıyor

Bir uygulama programının MQGET çağrısında **MQGMO_MARK_SKIP_BACKOUT** seçeneğini belirterek bir *MQGET-error-backout* döngüye girmesini önleyebilirsiniz.

Not: Yalnızca z/OSiçin WebSphere MQ ' da desteklenir.

Bir uygulama programı, bir iş biriminin bir parçası olarak, kuyruktan ileti almak için bir ya da daha fazla MQGET çağrısını yayınlayabilir. Uygulama programı bir hata saptarsa, iş birimini geri alabilirler. Bu işlem, o

iş birimi sırasında güncellenen tüm kaynakları, çalışma birimi başlatılmadan önce bulunduğu duruma geri yükler ve MQGET çağrıları tarafından alınan iletileri yeniden yürürlüğe kaydeder.

Bu iletiler, yeniden yürürlüğe girdikten sonra, uygulama programı tarafından yayınlanan sonraki MQGET çağrılarında kullanılabilir. Birçok durumda, bu durum uygulama programı için sorun yaratmaz. Ancak, geriletme sırasında ortaya çıkan hatanın çevresini geçersiz kılamaması durumunda, kuyrukta iletinin yeniden yürürlüğe girmesi uygulama programının bir *MQGET-error-backout* döngüye girmesine neden olabilir.

Bu sorunu önlemek için, MQGET çağrısında MQGMO_MARK_SKIP_BACKUT seçeneğini belirtin. Bu, uygulama tarafından başlatılan geriletme içine dahil edilmediği için MQGET isteğini işaretler; yani, yedeklenmemesi gerekir. Bu seçeneğin kullanılması, bir geri alma gerçekleştiğinde diğer kaynaklara ilişkin güncellemelerin gerektiği gibi yedekleneceği anlamına gelir, ancak işaretli iletinin yeni bir iş birimi altında alınmış gibi işlem göreceği anlamına gelir.

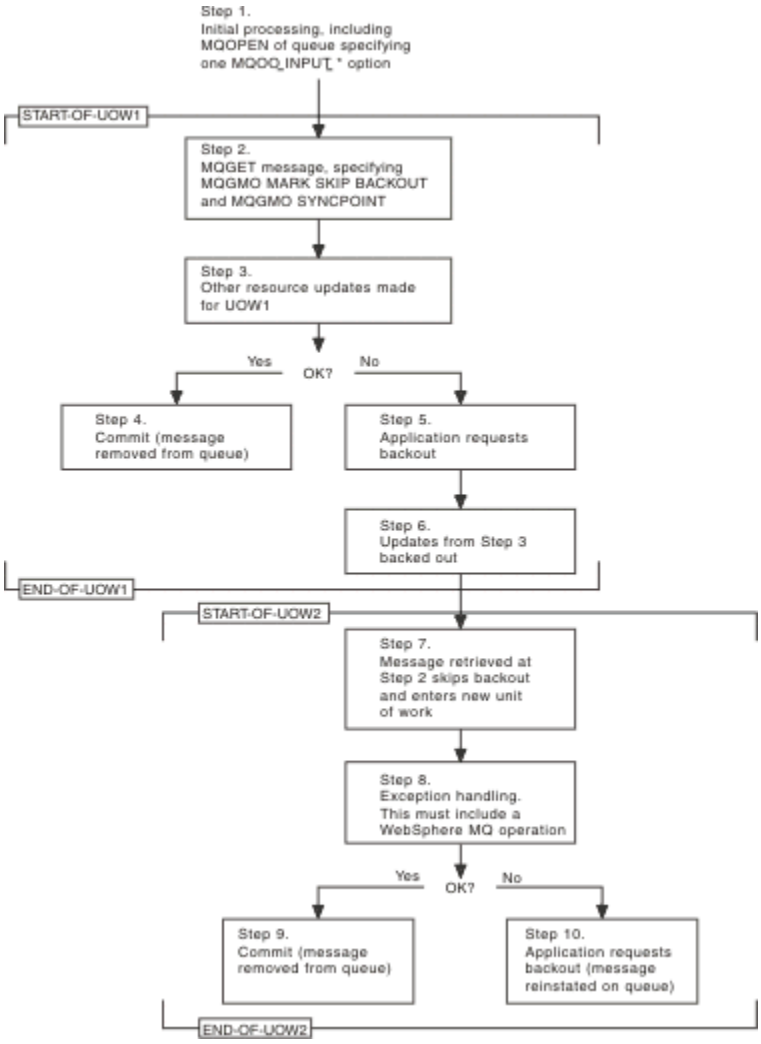
Uygulama programı, yeni iş birimini kesinleştirmek ya da yeni iş birimini yedeklemek için bir WebSphere MQ çağrısı yayınlamalıdır. Örneğin, program, iletiyi başlatan anda iletinin atıldığını bildiren bir kural dışı durum işleme işlemini gerçekleştirebilir ve iletiyi kuyruktan kaldırmak için iş birimini kesinleştirebilir; yeni iş birimi geriletirse (herhangi bir nedenle) ileti kuyruktan yeniden yürürlüğe girilir.

Bir iş birimi içinde, geri alma işlemi atlanıyor olarak işaretlenen yalnızca bir MQGET isteği olabilir; ancak, geri alma işlemi atlanıyor olarak işaretlenmemiş başka iletiler de olabilir. Bir ileti atlanıyor olarak imlendikten sonra, iş birimi içinde MQGMO_MARK_SKIP_BACKUT değerini belirten tüm MQGET çağrıları, MQRC_SECOND_MARK_NOT_ALLOWED neden koduyla başarısız olur.

Not:

1. İmli ileti, yalnızca bu iletiyi içeren iş birimi, bir uygulama isteği tarafından sona erdirildiyse yedeklenir. İş birimi başka herhangi bir nedenle yedeklendiyse, ileti, geri alma işlemini atlamak için işaretlenmemiş olması durumunda, kuyruğun üzerinde aynı şekilde yedeklenir.
2. Atlama geri alma işlemi, RRS tarafından denetlenen iş birimlerine katılan DB2 saklanmış yordamları içinde desteklenmez. Örneğin, MQGMO_MARK_SKIP_BACKUT seçeneğiyle bir MQGET çağrısı başarısız olur ve neden kodu MQRC_OPTION_ENVIRONMENT_ERROR ile başarısız olur.

Şekil 36 sayfa 258 , geri alma işlemini atlamak için bir MQGET isteği gerekirken uygulama programının içerebileceği tipik bir adım dizisi gösterir.



Şekil 36. MQGMO_MARK_SKIP_BACKUT kullanılarak geriletme atlanıyor

Şekil 36 sayfa 258 içindeki adımlar şunlardır:

1. Adım
İlk işlem, kuyruğun açılması için MQOPEN çağrısı da içinde olmak üzere, hareket içinde gerçekleşir (2. Adımdaki kuyruktan ileti almak için MQO_INPUT_ * seçeneklerinden birini belirtme).

2. Adım
MQGMO_SYNCPOINT ve MQGMO_MARK_SKIP_BACKUT ile MQGET çağrıldı. MQGMO_SYNCPOINT gereklidir; MQGET, MQGMO_MARK_SKIP_BACKUT için bir iş birimi içinde olmalıdır. Şekil 36 sayfa 258 içinde bu iş birimi UOW1olarak adlandırılır.

Adım 3
Diğer kaynak güncellemeleri UOW1' in bir parçası olarak yapılır. Bu, MQGET çağrılarını daha da içerebilir (MQGMO_MARK_SKIP_BACKUT olmadan verilir).

Adım 4
2. ve 3. adımlardaki tüm güncellemeler gerektiği şekilde tamamlar. Uygulama programı güncellemeleri kesinleştirir ve UOW1 sona erer. Adım 2 'de alınan ileti kuyruktan kaldırılır.

Adım 5
2. ve 3. adımlardaki güncellemelerin bazıları gerektiği gibi tamamlanmaz. Uygulama programı, bu adımlar sırasında yapılan güncellemelerin yedekleneceğini ister.

Adım 6
Adım 3 'te yapılan güncellemeler geriletilir.

Adım 7
2. Adımda yapılan MQGET isteği arka arkaya atlanır ve yeni bir iş biriminin (UOW2) bir parçası olur.

Adım 8
UOW2 , yedeklenmekte olan UOW1 ' a yanıt olarak kural dışı durum işlemeyi gerçekleştirir. (Örneğin, başka bir kuyruğa MQPUT çağrısı, UOW1 ' in yedeklenmesine neden olan bir sorunun ortaya çıktığını gösterir.)

Adım 9
Adım 8 gerektiği şekilde tamamlanır, uygulama programı etkinliği kesinleştirir ve UOW2 sona erer. MQGET isteği UOW2 ' nin bir parçası olduğundan (Adım 7 'ye bakın), bu kesinleştirme iletinin kuyruktan kaldırılmasına neden olur.

Adım 10
Adım 8 gerektiği gibi tamamlanmaz ve uygulama programı UOW2' yi yedeklemektedir. İleti alma isteği UOW2 ' nin bir parçası olduğundan (bkz. Adım 7), o da geriletilir ve kuyruğun yeniden yürürlüğe girmesini sağlar. Artık, bu ya da başka bir uygulama programı tarafından yayınlanan MQGET çağrılarını (kuyrukta başka bir iletiyle aynı şekilde) başka bir uygulama programı tarafından yayınlanabilir.

Uygulama verileri dönüştürme

Gerektiğinde, ileti tanımlayıcısı ve üstbilgi verilerini gerekli karakter kümesine ve kodlamaya dönüştüren MCA ' lar. Bağlantının her iki ucu (yani, yerel MCA ya da uzak MCA) dönüştürmeyi yapabilir.

Bir uygulama bir kuyruğa ileti yerleştirdiğinde, kuyruk yöneticileri ve MCA ' lar tarafından işlendiklerinde iletilerin denetimini kolaylaştırmak için, yerel kuyruk yöneticisi ileti tanımlayıcılara denetim bilgileri ekler. Ortama bağlı olarak, ileti üstbilgisi veri alanları, yerel sistemin karakter kümesi ve kodlamasında yaratılır.

Sistemler arasında ileti taşıdığınızda, bazen uygulama verilerini giriş sisteminin gerektirdiği karakter kümesiyle ve kodlamaya dönüştürmeniz gerekir. Bu işlem, alma sistemindeki uygulama programlarından ya da gönderme sistemindeki MCA ' lar tarafından yapılabilir. Alma sisteminde veri dönüştürme destekleniyorsa, gönderme sisteminde önceden ortaya çıkan dönüştürmeye bağlı olarak, uygulama verilerini dönüştürmek için uygulama programlarını kullanın.

Uygulama verileri bir uygulama programı içinde dönüştürülürken, MQGMO yapısının *Options* alanında MQGET çağrısına aktarılan MQGMO_CONVERT seçeneği belirtildiğinde ve *tüm* ' in doğru olduğu belirtilmektedir:

- The *CodedCharSetId* or *Encoding* fields set in the MQMD structure associated with the message on the queue differ from the *CodedCharSetId* or *Encoding* fields set in the MQMD structure specified on the MQGET call.
- İletiyle ilişkilendirilen MQMD yapısındaki *Format* alanı MQFMT_NONE değil.
- MQGET çağrısında belirtilen *BufferLength* sıfır değil.
- İleti verisi uzunluğu sıfır değil.
- Kuyruk yöneticisi, iletiyle ilişkili MQMD yapılarında belirtilen *CodedCharSetId* ve *Encoding* alanları arasındaki dönüştürmeyi destekler ve MQGET çağrısını destekler. Desteklenen kodlanmış karakter takımı tanıtıcıları ve makine kodlamalarıyla ilgili ayrıntılar için CodedCharSetId ve Encoding başlıklı konuya bakın.
- Kuyruk yöneticisi ileti biçiminin dönüştürülmesini destekler. İletiyle ilişkilendirilen MQMD yapısının *Format* alanı, yerleşik biçimlerden biriyse, kuyruk yöneticisi iletiyi dönüştürebilir. *Format* yerleşik biçimlerden biri değilse, iletiyi dönüştürmek için bir veri dönüştürme çıkışı yazmanız gerekir.

MCA ' yı gönderme işlemi verileri dönüştürecekse, dönüştürmenin gerekli olduğu her bir gönderenin ya da sunucu kanalının tanımında CONVERT (YES) anahtar sözcüğünü belirleyin. Veri dönüştürme başarısız olursa, ileti gönderen kuyruk yöneticisinde DLQ ' ya gönderilir ve MQDLH yapısının *Feedback* alanı nedeni gösterir. İleti DLQ ' ya (DLQ) yerleştirilemiyorsa, kanal kapanır ve dönüştürülemez ileti iletim kuyruğunda kalır. MCA ' ları göndermek yerine, uygulamalar içinde veri dönüştürme işlemi bu durumdan kaçınıyor.

Kural olarak, yerleşik biçim ya da veri dönüştürme çıkışı tarafından *karakter* olarak tanımlanan iletiden alınan veriler, ileti tarafından istenen ileti tarafından kullanılan kodlanmış karakter kümesinden ve *sayısal* alanlar, istenen kodlamaya dönüştürülmektedir.

Yerleşik biçimler dönüştürülürken kullanılan dönüştürme işleme kurallarına ilişkin ek ayrıntılar için ve kendi veri dönüştürme çıkışlarınızı yazmaya ilişkin bilgi için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397. Dil desteği tablolarıyla ve desteklenen makine kodlamalarıyla ilgili bilgi için ayrıca bkz. Ulusal diller ve Makine kodlamaları .

EBCDIC yeni satır karakterlerinin dönüştürülmesi

EBCDIC platformundan ASCII ' ye gönderdiğiniz verilerin yeniden geri aldığınız verilerle özdeş olduğundan emin olmanız gerekiyorsa, EBCDIC yeni satır karakterlerini dönüştürmeyi denetlemeniz gerekir.

Bunu, WebSphere MQ ' u değiştirmemiş dönüştürme tablolarını kullanacak şekilde zorlayan, platforma bağlı bir anahtar kullanarak yapabilirsiniz, ancak sonuçlanabilen tutarsız davranışlardan haberdar olmalısınız.

EBCDIC yeni satır karakteri, altyapılarda ya da dönüştürme çizelgelerinde tutarlı olarak dönüştürülmediği için sorun ortaya çıkar. Sonuç olarak, veriler bir ASCII altyapısında görüntüleniyorsa, biçimlendirme yanlış olabilir. Örneğin, RUNMQSC kullanan bir ASCII altyapısından bir IBM i sistemini uzaktan denetlemek, örneğin, bu işlemi zorlaştırır.

EBCDIC biçimli verileri ASCII biçimine dönüştürmeye ilişkin ek bilgi için Veri dönüştürme başlıklı konuya bakın.

Kuyruklardaki Iletilere Göz Atma

MQGET çağrısını kullanarak kuyruklardaki iletilere göz atmaya ilişkin bilgileri bulmak için bu bilgileri kullanın.

Bir kuyruktaki iletilere göz atmak için MQGET çağrısını kullanmak için:

1. MQOO_BROWSE seçeneğini belirterek, göz atma için kuyruğu açmak için MQOPEN ' i çağırın.
2. Kuyruktaki ilk iletiye göz atmak için, MQGET ' i MQGMO_BROWSE_FIRST seçeneğiyle çağırın. İstediğiniz iletiyi bulmak için, birçok iletiyi adımlamak için MQGET ' i MQGMO_BROWSE_NEXT seçeneğiyle sürekli olarak çağırın.

   MQMD yapısının *MsgId* ve *CorrelId* alanlarını, tüm iletileri görmek için MQGET çağrısından sonra boş değer olarak ayarlamanız *gerekir* .
3. Kuyruğu kapatmak için MQCLOSE ' yi çağırın.

Göz at imleci

Göz atma için bir kuyruk açtığınızda (MQOPEN), arama, göz atma seçeneklerinden birini kullanan MQGET çağrılarıyla kullanılmak üzere bir göz atma imleçle oluşturur. Göz atma imlecini, kuyruğun ilk iletisine göre konumlandırılmış bir mantıksal gösterge olarak düşünebilirsiniz.

Aynı kuyruk için birden çok MQOL isteği yayınlayarak, birden çok göz atma imleciniz (tek bir programdan) olabilir.

Göz atma için MQGET ' i aradığınızda, MQGMO yapılarınızda aşağıdaki seçeneklerden birini kullanın:

MQGMO_BROWSE_FIRST
MQMD yapılarınızda belirtilen koşulları karşılayan ilk iletinin bir kopyasını alır.

MQGMO_BROWSE_NEXT
MQMD yapılarınızda belirtilen koşulları karşılayan bir sonraki iletinin bir kopyasını alır.

MQGMO_BROWSE_MSG_UNDER_CURSOR
İmlecin şu anda gösterdiği iletinin bir kopyasını alır. Bu, en son MQGMO_BROWSE_FIRST ya da MQGMO_BROWSE_NEXT seçeneğini kullanarak alınan iletinin bir kopyasını alır.

Tüm durumlarda, ileti kuyrukta kalır.

Bir kuyruğu açtığınızda, imleç, kuyruktaki ilk iletiden hemen önce mantıksal olarak konumlandırılır. Bu, MQGET çağrısından hemen sonra MQGET çağrısını yaparsanız, ilk iletiye göz atmak için MQGMO_BROWSE_NEXT seçeneğini kullanabilirsiniz; MQGMO_BROWSE_FIRST seçeneğini kullanmanız gerekmez.

İletilerin kuyruktan kopyalandığı sıra, kuyruğun *MsgDeliverySequence* özniteliğinden belirlenir. (Daha fazla bilgi için bkz. “İletilerin kuyruktan alınma sırası” sayfa 235.)

FIFO ' daki (ilk giren, ilk çıkar) kuyruklar

Kuyrukta kuyrukta bulunan ilk ileti kuyrukta en uzun olan iletidir.

Kuyrukta sıralı olarak iletileri okumak için MQGMO_BROWSE_NEXT kullanın. Bu sırayla bir kuyruk olarak, kuyruğa göz atarken, kuyrukta kuyruğa yerleştirdiğiniz iletiler de sona ermiş olur. İmleç, kuyruğun sonuna ulaştığını algıladığında, imleç bulunduğu yerde kalır ve MQRC_NO_MSG_AVAILEABLE ile geri döndürür. Bundan sonra iletiyi daha fazla ileti bekliyor ya da MQGMO_BROWSE_FIRST çağrısıyla kuyruğun başlangıcınına sıfırlayabilirsiniz.

Öncelik sırasına göre kuyruklar

Kuyrukta kuyrukta bulunan ilk ileti, kuyrukta en uzun süredir yer alan ve MQOPEN çağrısının verildiği sırada en yüksek önceliğe sahip olan iletidir.

Kuyruktaki iletileri okumak için MQGMO_BROWSE_XT seçeneğini kullanın.

Göz atma imleci, ilk iletinin önceliğiyle en düşük önceliğe kadar olan iletinin önceliğiyle çalışarak sonraki iletiyi işaret eder. Bu süre içinde, yürürlükteki göz atma imlecinin belirlediği iletiye, bu süre boyunca kuyruğa konması gereken iletiler, bu süre içinde kuyruğa konması ya da daha düşük olduğu sürece göz atabilir.

Daha yüksek önceliğe sahip kuyruğa gönderilen iletiler yalnızca aşağıdaki şekilde göz atılabilir:

- Yeniden göz atma için kuyruğu açma işlemi, yeni bir göz atma imlecinin kurulduğu noktadır.
- MQGMO_BROWSE_FIRST seçeneğinin kullanılması

Kesinleştirilmemiş iletiler

Kesinleştirilmemiş bir ileti bir göz atma için hiçbir zaman görünür değildir; göz atma imleci geçmişteki atlamadır.

Bir iş birimi içindeki iletiler, iş birimi kesinleştirilinceye kadar göz atılamaz. İletiler, kesinleştirildiğinde kuyruklardaki konumlarını değiştirmez; bu nedenle, MQGMO_BROWSE_FIRST seçeneğini kullanmazsanız ve kuyruk yeniden çalışsa da, kesinleştirilmemiş iletiler görülmez, kesinleştirilmemiş iletiler *işlenmez* .

Kuyruk sırasına değiştir

Kuyrukta ileti varken, ileti teslimi işlemi öncelikten FIFO ' ya çevrilirse, kuyruğa yollanan iletilerin sırası değiştirilmez. Kuyruğa daha sonra eklenen iletiler, kuyruğun varsayılan önceliğini alır.

Kuyruğun dizinini kullanma

Yalnızca tek bir öncelik (kalıcı ya da kalıcı olmayan ya da her ikisi) iletileri içeren dizinlenmiş bir kuyruğa göz attığınızda, kuyruk yöneticisi belirli göz atma formlarının kullanıldığı zamanlara göz atmak için dizini kullanır.

Not: Yalnızca z/OSiçin WebSphere MQ ' da desteklenir.

Dizinlenen bir kuyruk yalnızca tek önceliğe ilişkin iletiler içerdiğinde, aşağıdaki göz atma biçimlerinden herhangi biri kullanılır:

1. Kuyruk MSGID tarafından dizinlendiyse, hedef iletiyi bulmak için dizin kullanılarak MQMD yapısındaki bir MSGID geçiren isteklere göz atın.
2. Kuyruk, CORRELID tarafından dizinlendiyse, hedef iletiyi bulmak için dizin kullanılarak MQMD yapısındaki bir CORRELID geçiren isteklere göz atın.
3. Kuyruk GROUPID tarafından dizinlendiyse, hedef iletiyi bulmak için, dizin kullanılarak MQMD yapısındaki bir GROUPID geçiren isteklere göz atın.

Göz atma isteği MQMD yapısındaki bir MSGID, CORRELID ya da GROUPID iletmiyorsa, kuyruk dizinlenir ve bir ileti döndürülür, ileti için dizin girdisi bulunmalı ve bu ileti içindeki bilgiler, göz atma imlecini güncelleştirmek için kullanılır. Dizin değerleri için geniş bir seçim kullanırsanız, bu değer göz atma isteğine önemli bir ek işlem eklemez.

İleti uzunluğu bilinmediği zaman iletilere göz atmanızı sağlar

İletinin boyutunu bilmediğinizde bir iletiye göz atmak ve iletiyi bulmak için *MsgId*, *CorrelId*ya da *GroupId* alanlarını kullanmak istemiyorsanız, MQGMO_BROWSE_MSG_UNDER_CURSOR seçeneğini kullanabilirsiniz:

1. Bununla birlikte bir MQGET komutu verin:
   - MQGMO_BROWSE_FIRST ya da MQGMO_BROWSE_NEXT seçeneği
   - MQGMO_ACCEPT_TRUNCATED_MSG seçeneği
   - Arabellek uzunluğu sıfır

   Not: Başka bir program da aynı iletiyi alacaksa, MQGMO_LOCK seçeneğini de kullanmayı düşünün. MQRC_TRUNCATED_MSG_ACCEPTED geri döndürülemelidir.
2. Gereken depolama alanını ayırmak için döndürülen *DataLength* ' i kullanın.
3. MQGMO_BROWSE_MSG_UNDER_CURSOR ile bir MQGET yayınlayın.

Alınan ileti son olarak alındı; göz atma imleci hareket ettirilmedi. MQGMO_LOCK seçeneğini kullanarak iletiyi kilitlemek ya da MQGMO_UNLOCK seçeneğini kullanarak kilitli bir iletinin kilidini açmak için bu seçeneği belirleyebilirsiniz.

Kuyruk açıldığından, MQGMO_BROWSE_FIRST ya da MQGMO_BROWSE_NEXT seçeneklerinde MQGET işlemi yapılmazsa, çağrı başarısız olur.

Göz attığınız bir iletinin kaldırılması

Kuyruktan, iletileri kaldırmak için ve göz atılmasına ilişkin kuyruğu açtığınız için, önceden göz attığınız bir ileti kuyruğundan kaldırabilirsiniz. (MQOO_Browse seçeneğinin yanı sıra, MQOO_INPUT_ * seçeneklerinden birini ve MQOO_Browse seçeneğini belirtmelisiniz.)

İletiyi kaldırmak için, MQGET ' yi yeniden çağırın, ancak MQGMO yapısının *Options* alanında MQGMO_MSG_UNDER_CURSOR değerini belirtin.Bu durumda, MQGET çağrısı MQMD yapısındaki *MsgId*, *CorrelId*ve *GroupId* alanlarını yoksayar.

Göz atma ve kaldırma adımlarınız arasında, başka bir program kuyruktan iletileri kaldırmış olabilir; bu iletiler göz atma imleçinizin altındaki ileti de içinde olmak üzere, iletileri kaldırmış olabilir. Bu durumda, MQGET çağrıınız, iletinin kullanılamadığını bildiren bir neden kodu döndürür.

Mantıksal düzende iletilere göz atma

“Mantıksal ve fiziksel sıralama” sayfa 235 , kuyruklardaki iletilerin mantıksal ve fiziksel sırası arasındaki farkı açıklar. Bu ayrım özellikle bir kuyruğa göz atılırken önemlidir; çünkü, genel olarak, iletiler silinmez ve göz atma işlemleri kuyruğun başlangıcında başlamalarına da gerek yoktur.

If an application browses through the various messages of one group (using logical order), it is important that logical order should be followed to reach the start of the next group, because the last message of one group might occur physically *bundan sonra* the first message of the next group. MQGMO_LOGICAL_ORDER seçeneği, bir kuyruk taranırken mantıksal sıraların izlenmesini sağlar.

Göz atma işlemleri için MQGMO_ALL_MSGS_AVALANABILIR (ya da MQGMO_ALL_SEGMENTS_AVALABILIR) kullanın. Mantıksal ileti vakasını MQGMO_ALL_MSGS_AVALABILIR ile göz önünde bulundurun. Bunun etkisi, mantıksal bir iletinin yalnızca gruptaki kalan tüm iletilerin da mevcut olması durumunda kullanılabilir olması. Eğer değillerse, mesaj iletilir. Bu, eksik iletiler daha sonra geldiğinde, bir sonraki işlem tarafından fark edilmemeleri anlamına gelebilir.

Örneğin, aşağıdaki mantıksal iletiler mevcutsa,

```
   Logical message 1 (not last) of group 123
   Logical message 1 (not last) of group 456
   Logical message 2 (last)     of group 456
```

ve bir göz atma işlevi MQGMO_ALL_MSGS_AVALABILIR ile verilir, 456 grubunun ilk mantıksal iletisi döndürülür; bu mantıksal iletiye göz atma işlevi bırakılır. 123 grubunun ikinci (son) iletisi şimdi varırsa:

```
   Logical message 1 (not last) of group 123
   Logical message 2 (last)     of group 123
   Logical message 1 (not last) of group 456  <=== browse cursor
   Logical message 2 (last)     of group 456
```

ve aynı gözatma işlevi yayınlansa da, bu grubun ilk iletisi *önce* olduğu için 123 grubunun artık tamamlanmış olduğunu fark etmemektedir.

Bazı durumlarda (örneğin, grup tümüyle yok edici olarak alındıysa), MQGMO_ALL_MSGS_AVAM değerini önce MQGMO_BROWSE_FIRST ile birlikte kullanabilirsiniz. Tersi durumda, gözden kaçan yeni gelen iletileri not almak için taramayı yinelemeniz gerekir; MQGMO_BROWSE_NEXT ve MQGMO_ALL_MSGS_AVAILLY ile birlikte MQGMO_BEKE yayınının verilmesi, bu iletilerin dikkate almadığını gösterir. (Bu durum, iletilerin taranmasından sonra varabilecek daha yüksek öncelikli iletilere de oluşur.)

Sonraki bölümler, kesimlere ayrılmış iletiler ile ilgili göz atma örneklerine göz atmanızı sağlar; bölümlenmiş iletiler benzer ilkeleri izler.

Gruplardaki iletilere göz atma
Bu örnekte, uygulama kuyrukta, mantıksal sırada her bir iletiyle göz atılıyor.

Kuyruktaki iletiler gruplandırılmış olabilir. Gruplanmış iletiler için uygulama, içindeki tüm iletiler gelene kadar, herhangi bir grubu işlemeye başlamak istemiyor. Bu nedenle, gruptaki ilk ileti için MQGMO_ALL_MSGS_AVALABILIR belirtildi; gruptaki sonraki iletiler için bu seçenek gereksiz.

Bu örnekte MQGMO_WAN kullanıldı. However, although the wait can be satisfied if a new group arrives, for the reasons in “Mantıksal düzende iletilere göz atma” sayfa 262, it is not satisfied if the browse cursor has already passed the first logical message in a group, and the remaining messages now arrive. Bununla birlikte, uygun bir aralık beklemek, yeni iletiler ya da kesimler beklenirken uygulamanın sürekli olarak döngüye girmemesini sağlar.

MQGMO_LOGICAL_ORDER, taramanın mantıksal sırada olduğundan emin olmak için kullanılır. Bu yıkıcı MQGET örneğiyle, her grubun kaldırıldığı için, bir gruptaki ilk (ya da tek) iletiyi ararken MQGMO_LOGICAL_ORDER kullanılmaz.

Uygulamanın arabelleğinin tüm iletiyi tutabilmek için her zaman yeterince büyük olduğu varsayılır. İletinin bölümlenip kesilmediği varsayılır. Bu nedenle, her MQGET üzerinde MQGMO_COMPLE_MSG belirtildi.

Aşağıda, bir gruptaki mantıksal iletilere göz atmanın bir örneği verilmektedir:

```
/* Browse the first message in a group, or a message not in a group */
GMO.Options = MQGMO_BROWSE_NEXT | MQGMO_COMPLETE_MSG | MQGMO_LOGICAL_ORDER
            | MQGMO_ALL_MSGS_AVAILABLE | MQGMO_WAIT
MQGET GMO.MatchOptions = MQMO_MATCH_MSG_SEQ_NUMBER, MD.MsgSeqNumber = 1
/* Examine first or only message */
 ...

GMO.Options = MQGMO_BROWSE_NEXT | MQGMO_COMPLETE_MSG | MQGMO_LOGICAL_ORDER
do while ( GroupStatus == MQGS_MSG_IN_GROUP )
```

```
   MQGET
   /* Examine each remaining message in the group */
   ...
```

Grup, MQRC_NO_MSG_AVAILEABLE iade edilinceye kadar yinelenir.

İmha edici olarak göz atma ve alma
Bu örnekte uygulama, grup içindeki her bir mantıksal iletiyi, bu grubun yok edici bir şekilde almaya karar vermeden önce göz önünde bulunmaya devam eder.

Bu örneğe ilişkin ilk bölüm öncekine benzer. Ancak, bu durumda, bütün bir grubu göz altında bulunca, geri dönüp onu yok edici bir şekilde geri almaya karar veriyoruz.

Bu örnekte her grup kaldırıldığı için, bir gruptaki ilk ya da tek ileti aranırken MQGMO_LOGICAL_ORDER kullanılmaz.

Aşağıda bir göz atma örneği verilir ve daha sonra yok edici duruma gelir elde edin:

```
GMO.Options = MQGMO_BROWSE_NEXT | MQGMO_COMPLETE_MSG | MQGMO_LOGICAL_ORDER
            | MQGMO_ALL_MESSAGES_AVAILABE | MQGMO_WAIT
do while ( GroupStatus == MQGS_MSG_IN_GROUP )
   MQGET
   /* Examine each remaining message in the group (or as many as
      necessary to decide whether to get it destructively) */
   ...

if ( we want to retrieve the group destructively )

   if ( GroupStatus == ' ' )
      /* We retrieved an ungrouped message */
      GMO.Options = MQGMO_MSG_UNDER_CURSOR | MQGMO_SYNCPOINT
      MQGET GMO.MatchOptions = 0
      /* Process the message */
      ...

   else
      /* We retrieved one or more messages in a group.  The browse cursor */
      /* will not normally be still on the first in the group, so we have */
      /* to match on the GroupId and MsgSeqNumber = 1.                    */
      /* Another way, which works for both grouped and ungrouped messages,*/
      /* would be to remember the MsgId of the first message when it was  */
      /* browsed, and match on that.                                      */
      GMO.Options = MQGMO_COMPLETE_MSG | MQGMO_SYNCPOINT
      MQGET GMO.MatchOptions = MQMO_MATCH_GROUP_ID
                             | MQMO_MATCH_MSG_SEQ_NUMBER,
           (MQMD.GroupId      = value already in the MD)
            MQMD.MsgSeqNumber = 1
      /* Process first or only message */
      ...

      GMO.Options = MQGMO_COMPLETE_MSG | MQGMO_SYNCPOINT
                  | MQGMO_LOGICAL_ORDER
      do while ( GroupStatus == MQGS_MSG_IN_GROUP )
         MQGET
         /* Process each remaining message in the group */
         ...
```

Browsed iletilerinin sürekli olarak sunulmasını önleme

Belirli açık seçenekleri ve ileti alma seçeneklerini kullanarak, iletileri geçerli ya da diğer işbirliği uygulamaları tarafından yeniden alınmamaları için göz atıldığı gibi işaretleyebilirsiniz. İletiler, göz atma için açık bir şekilde ya da otomatik olarak yeniden kullanılabilir duruma getirmek için otomatik olarak işaretleyemez.

Bir kuyruktaki iletilere göz atsanız, onları yok edici bir şekilde elde etseniz, bunları alabileceğiniz sıraya göre farklı bir sırayla alabilirsiniz. Özellikle, aynı iletiye birden çok kez göz atabilirsiniz; bu, kuyruktan kaldırılsa da olanaklı değildir. Bundan kaçınmak için, iletileri göz attıkları gibi *işaretleyebilir* ve işaretli iletileri almayı önlemeniz gerekir. Bu, bazen *işaretle göz at*olarak da adlandırılır. Göz atılan iletileri işaretlemek için, MQGMO_MARK_BROWSE_HANDLE ileti alma seçeneğini kullanın ve yalnızca işaretlenmemiş iletileri almak için MQGMO_UNMARKET_BROWSE_MSG kullanın. MQGMO_BROWSE_FIRST, MQGMO_UNMARKED_BROWSE_MSG ve MQGMO_MARK_BROWSE_HANDLE seçeneklerini ve yinelenen MQGES komutunu kullanırsanız, kuyrukta bulunan her iletiyi sırayla alırsınız.

Bu, iletilerin atlanmamasını sağlamak için MQGMO_BROWSE_FIRST kullanılsa da iletilerin yinelenmesini önler. Bu seçenekler birleşimi, tek bir sabit MQGMO_BROWSE_HANDLE ile gösterilebilir. Kuyruğunda göz atılmamış ileti olmadığında, MQRC_NO_MSG_AVAILEABLE iletisi döndürülür.

Birden çok uygulama aynı kuyruğa göz atıyorsa, kuyrukları MQOO_CO_OP ve MQOO_BROOK seçenekleriyle açabilir. Her bir MQOPER tarafından döndürülen nesne tanıtıcısı, işbirliği yapan grubun bir parçası olarak kabul edilir. MQGMO_MARK_BROWSE_CO_OP seçeneğini belirten bir MQGET çağrısının döndürdüğü herhangi bir ileti, bu işbirliği işlemi kümesi için işaretlendi olarak kabul edilir.

Bir ileti bir süredir imlendiyse, kuyruk yöneticisi tarafından otomatik olarak imlenemez ve göz atmak için kullanılabilir kılınabilir. Kuyruk yöneticisi özniteliği MsgMarkBrowseInterval , iş birliği tanıtıcısı olarak iş birliği yapmak için bir iletinin gösterileceği süreyi milisaniye cinsinden verir. Bir MsgMarkBrowseInterval /-1, iletilerin hiçbir zaman otomatik olarak işaretlenmediği anlamına gelir.

Tek bir işlem ya da işbirliği süreci kümesi iletileri durdururken, işaretlenen iletiler işaretsiz olur.

İşbirliğine göz atma örnekleri

Bir kuyruktaki iletilere göz atmak ve her iletinin içeriğine dayalı olarak bir tüketici başlatmak için bir dağıtıcı uygulamasının birden çok kopyasını çalıştırabilirsiniz. Her dağıtıcıda, kuyruğun MQOO_CO_OP ile birlikte açılmasını sağlar. Bu, dağıtıcıların işbirliği yaptığını ve birbirlerinin işaretlenen iletilerinden haberdar olacağını gösterir. Daha sonra, MQGMO_BROWSE_FIRST, MQGMO_UNMARKET_BROWSE_MSG ve MQGMO_MARK_BROWSE_CO_OP seçeneklerini belirterek, her dağıtıcı yinelenen MQGET çağrıları yapar (bu seçenek bileşimini göstermek için tek bir sabit MQGMO_BROWSE_CO_OP kullanabilirsiniz). Daha sonra, her dağıtıcı uygulaması yalnızca başka işbirliği dağıtıcıları tarafından işaretlenmemiş iletileri alır. Dağıtıcı, bir tüketici başlatır ve MQGET tarafından döndürülen MsgToken ' ı, iletiyi yok edici olarak kuyruktan alan tüketiciye iletir. Tüketici iletinin MQGET ' ini yedeklerse, bu ileti artık işaretlenmediği için, tarayıcılardan biri için yeniden gönderimi yapmak için ileti kullanılabilir. Tüketici iletide bir MQGET işlemi yapmazsa, MsgMarkBrowseInterval iletildikten sonra kuyruk yöneticisi, iş birliği yapan tutamaçlar kümesi için iletiyi işaretler ve yeniden dağıtılabilir.

Aynı dağıtıcı uygulamasının birden çok kopyası yerine, kuyruğa göz atan, kuyruklardaki iletilerin bir alt kümesini işlemek için uygun olan farklı dağıtıcı uygulamalarından birine sahip olabilirsiniz. Her dağıtıcıda, kuyruğun MQOO_CO_OP ile birlikte açılmasını sağlar. Bu, dağıtıcıların işbirliği yaptığını ve birbirlerinin işaretlenen iletilerinden haberdar olacağını gösterir.

- Tek bir dağıtıcıya ilişkin ileti işleme sırası önemliyse, her dağıtıcı MQGMO_BROWSE_FIRST, MQGMO_UNMARKED_BROWSE_MSG ve MQGMO_MARK_BROWSE_HANDLE (ya da MQGMO_BROWSE_HANDLE) seçeneklerini belirterek, MQGET çağrılarını yineledi. Bu dağıtıcının işlemesi için browsed iletisi uygunsa, MQMO_Match_MSG_XX_ENCODE_CASE_CAPS_LOCK_ON token, mqgmo_mark_browse_co_op ve önceki MQGET çağrısının döndürdüğü MsgToken adlı bir MQGET çağrısı yapar. Arama başarılı olursa, dağıtıcı tüketiciyi kullanıma hazırlar ve MsgToken ' ı buna iletir.
- İleti işleme sırası önemli değilse ve dağıtıcının karşılaştığı iletilerin çoğunu işlemesi bekleniyorsa, MQGMO_BROWSE_FIRST, MQGMO_UNMARKED_BROWSE_MSG ve MQGMO_MARK_BROWSE_CO_OP (ya da MQGMO_BROWSE_CO_OP) seçeneklerini kullanın. Dağıtıcı işleyemediği bir iletiyi göz atarsa, MQGET komutunu MQMO_MATCH_MSG_TOKEN, MQGMO_UNMARK_BROWSE_CO_OP seçeneğiyle çağırarak, daha önce döndürülen MsgToken ile iletiyi işaretler.

MQGET çağrısının başarısız olduğu bazı durumlar

Bir kuyruğun belirli öznitelikleri bir MQOPER ve MQGET çağrısının verilmesi arasındaki bir komutta FORCE seçeneği kullanılarak değiştirilirse, MQGET çağrısı başarısız olur ve MQRC_OBJECT_CHANGED neden kodunu döndürür.

Kuyruk yöneticisi, nesne tanıtıcısını artık geçerli değil olarak işaretler. Bu durum, değişikliklerin kuyruk adının çözümlediği herhangi bir kuyruğa uygulanırsa da olur. Bu şekilde, tanıtıcıyı etkileyen öznitelikler, MQOPEN' da MQOPER çağrısının tanımında listelenir. Aramanız MQRC_OBJECT_CHANGED neden kodunu döndürürse, kuyruğu kapatın, yeniden açın ve bir ileti almaya yeniden çalışın.

İletileri alma girişiminde bulunduğunuz bir kuyruk için (ya da kuyruk adının çözümlendiği herhangi bir kuyruk) alma işlemleri engelleniyorsa, MQGET çağrısı başarısız olur ve MQRC_GET_INHIBITED neden

kodunu döndürür. Bu durum, göz atma için MQGET çağrısını kullanıyor olsanız bile ortaya çıktı. Daha sonra MQGET çağrısını denerseniz başarıyla bir ileti alabilirsiniz; uygulamanın tasarımı diğer programlar kuyrukların özniteliklerini düzenli olarak değiştiriyorsa, bu iletiyi başarıyla alabilirsiniz.

Dinamik bir kuyruk (geçici ya da kalıcı) silindiyse, önceden edinilmiş bir nesne tanıtıcısı kullanılarak MQGET çağrıları başarısız olur ve MQRC_Q_DELETED neden kodunu geri döndürür.

Yayınlama/abone olma uygulamaları yazılıyor

Yayınlama/abone olma WebSphere MQ uygulamalarını yazmaya başlayın.

Yayınlama/abone olma kavramlarına genel bakış için bkz. WebSphere MQ yayınlama/abone olma mesajlarına giriş.

Farklı yayınlama/abone olma uygulamaları yazılmasına ilişkin bilgi edinmek için aşağıdaki konulara bakın:

- "Yayınlayıcı uygulamaları yazılıyor" sayfa 267
- "Abone uygulamaları yazılıyor" sayfa 273
- "Yaşam çevrimlerini yayınla/abone ol" sayfa 289
- "İleti özelliklerini yayınla/abone ol" sayfa 293
- "İleti sıralaması" sayfa 295
- "Yayınların ele geçirmesi" sayfa 295
- "Yayın seçenekleri" sayfa 303
- "Abonelik seçenekleri" sayfa 303

İlgili kavramlar

"Uygulama geliştirme kavramları" sayfa 7
IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

"Kullanılacak programlama diline karar verme" sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

"IBM WebSphere MQ uygulamalarını tasarlama" sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

"Örnek WebSphere MQ programları" sayfa 92
Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

"Kuyruğa alma uygulaması yazılıyor" sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

"İstemci uygulamaları yazılıyor" sayfa 335
What you need to know to write client applications on WebSphere MQ.

" WebSphere MQ' da Web hizmetlerini kullanma" sayfa 907
SOAP için IBM WebSphere MQ iletimi ya da HTTP için IBM WebSphere MQ köprüsü kullanılarak Web hizmetleri için IBM WebSphere MQ uygulamaları geliştirebilirsiniz.

"IBM WebSphere MQ uygulaması oluşturulması" sayfa 409
Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

"Program hatalarının işlenmesi" sayfa 526
Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

Yayınlayıcı uygulamaları yazılıyor

İki örnek üzerinde çalışarak yayınlayıcı uygulamaları yazmaya başlayın. İlki, bir kuyruğa ileti yerleştirmek için bir noktadan noktaya mümkün olduğunca yakın bir şekilde modellenir ve ikinci olarak konular, yayınlayıcı uygulamaları için dinamik olarak daha yaygın bir kalıba yol açmaktadır.

Writing a simple WebSphere MQ publisher application is just like writing a WebSphere MQ point to point application that puts messages to a queue (Çizelge 41 sayfa 267). The difference is you MQPUT messages to a topic, not to a queue.

Çizelge 41. Yayınlama/yayınlama/abone olma WebSphere MQ program örüntülerini işaret eden nokta.

Adım	**Nokta MQ Çağrısı**	**MQ Call 'ı Yayınla**
Kuyruk yöneticisine bağlan	MQCONN	MQCONN
Kuyruğu aç	MQOPEN	
Konuyu aç		MQOPEN
İleti (lar) koy	MQPUT	MQPUT
Konuyu kapat		MQCLOSE
Kuyruğu kapat	MQCLOSE	
Kuyruk yöneticisinden bağlantıyı kes	MQDISC	MQDISC

Bu betonu yapmak için, hisse senedi fiyatlarını yayınlayabilmek için iki uygulama örneği vardır. İlk örnekte (“Örnek 1: Sabit bir konuya yayınlayıcı” sayfa 267), bir kuyruğa ileti yerleştirerek yakından modellenen yönetici, kuyruk yaratmak için benzer bir şekilde bir konu tanımlaması yaratır. MQPUT programlayıcı kodları, iletileri bir kuyruğa yazmak yerine konuya yazmak için kodlamayı sağlar. İkinci örnekte (“Örnek 2: Bir değişken konusuna yayınlayıcı” sayfa 270), programın WebSphere MQ ile olan etkileşiminin örünmesi benzerdir. Fark, programcının, yöneticinin yerine, iletinin yazıldığı konuyu yönetici yerine getirmektedir. Uygulamada genellikle, konu dizgisinin bir tarayıcı aracılığıyla insan girişi gibi başka bir kaynak tarafından ya da başka bir kaynak tarafından sağlandığı anlamına gelir.

İlgili kavramlar

“Abone uygulamaları yazılıyor” sayfa 273
Üç örnek üzerinde çalışarak abone uygulamaları yazmaya başlayın: Bir WebSphere MQ uygulaması, bir kuyruktan iletiler tüketiyor, abonelik yaratan ve kuyruğa alma hakkında bilgi sahibi olmamasını gerektiren bir uygulama ve son olarak hem kuyruklama hem de abonelikler kullanan bir örnek.

İlgili başvurular

KONUYU TANIMLA
KONUYU GÖRÜNTÜLE
TANıTıM

Örnek 1: Sabit bir konuya yayınlayıcı

Bir yönetimsel olarak tanımlanmış bir konuya yayınlama göstermek için bir WebSphere MQ programı.

Not: Sıkıştırılmış kodlama biçemi, üretim kullanımına hazır olmamaya yöneliktir.

See the output in Şekil 38 sayfa 268

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>
int main(int argc, char **argv)
{
    char     topicNameDefault[]   = "IBMSTOCKPRICE";
    char     publicationDefault[] = "129";
    MQCHAR48 qmName = "";

    MQHCONN  Hconn = MQHC_UNUSABLE_HCONN; /* connection handle                    */
    MQHOBJ   Hobj = MQHO_NONE;            /* object handle sub queue              */
    MQLONG   CompCode = MQCC_OK;          /* completion code                      */
    MQLONG   Reason = MQRC_NONE;          /* reason code                          */
    MQOD     td = {MQOD_DEFAULT};         /* Object descriptor                    */
    MQMD     md = {MQMD_DEFAULT};         /* Message Descriptor                   */
    MQPMO   pmo = {MQPMO_DEFAULT};        /* put message options                  */
    MQCHAR   resTopicStr[151];            /* Returned vale of topic string        */
    char *   topicName   = topicNameDefault;
    char *   publication = publicationDefault;
    memset   (resTopicStr, 0 , sizeof(resTopicStr));

    switch(argc){                         /* replace defaults with args if provided */
        default:
            publication = argv[2];
        case(2):
            topicName = argv[1];
        case(1):
            printf("Optional parameters:  TopicObject Publication\n");
    }
    do {
        MQCONN(qmName, &Hconn, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        td.ObjectType = MQOT_TOPIC;        /* Object is a topic                    */
        td.Version = MQOD_VERSION_4;       /* Descriptor needs to be V4            */
        strncpy(td.ObjectName, topicName, MQ_TOPIC_NAME_LENGTH);
        td.ResObjectString.VSPtr = resTopicStr;
        td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
        MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        pmo.Options = MQPMO_FAIL_IF_QUIESCING | MQPMO_RETAIN;
        MQPUT(Hconn, Hobj, &md, &pmo, (MQLONG)strlen(publication)+1, publication, &CompCode,
&Reason);
        if (CompCode != MQCC_OK) break;
        MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQDISC(&Hconn, &CompCode, &Reason);
    } while (0);
    if (CompCode == MQCC_OK)
        printf("Published \"%s\" using topic \"%s\" to topic string \"%s\"\n",
               publication, td.ObjectName, resTopicStr);
    printf("Completion code %d and Return code %d\n", CompCode, Reason);
}
```

Şekil 37. Basit WebSphere MQ yayıncısı sabit bir konuya yayındır.

```
X:\Publish1\Debug>PublishStock
Optional parameters:  TopicObject Publication
Published "129" using topic "IBMSTOCKPRICE" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0

X:\Publish1\Debug>PublishStock IBMSTOCKPRICE 155
Optional parameters:  TopicObject Publication
Published "155" using topic "IBMSTOCKPRICE" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0
```

Şekil 38. İlk yayınlayıcı örneğinden alınan örnek çıktı

Aşağıdaki seçilen kod satırları, WebSphere MQiçin bir yayınlayıcı uygulaması yazılmasına ilişkin konuları göstermektedir.

char topicNameDefault[] = "IBMSTOCKPRICE";
Programda varsayılan bir konu adı tanımlanıyor. Programa ilişkin ilk bağımsız değişken olarak farklı bir konu nesnesinin adını belirterek bu değeri geçersiz kılabilirsiniz.

MQCHAR resTopicStr[151];
`resTopicStr` is pointed at by `td.ResObjectString.VSPtr` and is used by `MQOPEN` to return the resolved topic string. Make the length of `resTopicStr` one larger than the length passed in `td.ResObjectString.VSBufSize` to give space for null termination.

memset (resTopicStr, 0, sizeof(resTopicStr));
Initialize `resTopicStr` to nulls to ensure the resolved topic string returned in an `MQCHARV` is null terminated.

td.ObjectType = MQOT_TOPIC
Yayınlama/abone olma için yeni bir nesne tipi vardır: *konu nesnesi*.

td.Version = MQOD_VERSION_4;
Yeni nesne tipini kullanmak için, nesne tanımlayıcısının en az *sürüm 4* ' ü kullanmanız gerekir.

strncpy(td.ObjectName, topicName, MQ_OBJECT_NAME_LENGTH);
`topicName` , bir konu nesnesinin adıdır; bazen de bir denetim konusu nesnesi olarak adlandırılır. In the example the topic object needs to be created beforehand, using WebSphere MQ Explorer or this MQSC command,

```
DEFINE TOPIC(IBMSTOCKPRICE) TOPICSTR(NYSE/IBM/PRICE) REPLACE;
```

td.ResObjectString.VSPtr = resTopicStr;
Çözümlenen konu dizesi, programda son `printf` ' da yankılanır. Çözümlenen dizgiyi programa geri döndürmek için WebSphere MQ için `MQCHARV ResObjectString` yapısını ayarlayın.

MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
Çıkış için konuyu açın; örneğin, çıkış kuyruğu açmak gibi.

pmo.Options = MQPMO_FAIL_IF_QUIESCING | MQPMO_RETAIN;
Yeni abonelerin yayını almasını ve yayınlayıcıda `MQPMO_RETAIN` belirtilmesini belirterek, abone başlatılmadan önce yayınlanan en son yayını alır ve ilk eşleşen yayınını alır. alternatifi ise abonelerin sadece abone başladıktan sonra yayınlanan yayınlarla sağlanmasıdır. Ek olarak bir abonede, abonelikte `MQSO_NEW_PUBLICATIONS_ONLY` belirterek alıkonan bir yayını alma seçeneği de reddedilir.

MQPUT(Hconn, Hobj, &md, &pmo, (MQLONG)strlen(publication)+1, publication, &CompCode, &Reason);
Add 1 to the length of the string passed to `MQPUT` to pass the null termination character to WebSphere MQ as part of the message buffer.

İlk örnek ne gösteriyor? The example imitates as closely as possible the tried and tested traditional pattern for writing point to point WebSphere MQ programs. WebSphere MQ programlama örüntüünün önemli bir özelliği, programcının iletilerin gönderileceği yerde ilgilenmemesidir. Programcının görevi, bir kuyruk yöneticisine bağlanmaktan ve alıcılara dağıtılacak olan iletileri iletmekten geçer. Noktadan noktaya paradigmada, programcı yöneticinin yapılandırdığı bir kuyruğu (büyük olasılıkla bir diğer ad kuyruğu) açar. Diğer ad kuyruğu, iletileri yerel kuyruk yöneticisinde ya da uzak bir kuyruk yöneticisinde hedef kuyruğa yöneltebiliyor. İletiler teslim edilmeyi beklerken, kaynak ve hedef arasında bir yerde kuyruklar üzerinde depolanır.

Yayınlama/abone olma örüntüünde, kuyruk açmak yerine, programcı bir konu açar. Örneğimizde konu, yönetici tarafından bir konu dizgisiyle ilişkilendirilir. Kuyruk yöneticisi, kuyrukları kullanarak yayını, yayının konu dizgisiyle eşleşen abonelikleri olan yerel ya da uzak abonelere iletir. Yayınlar alıkonsa, kuyruk yöneticisi artık abonesi olmasa da yayının en son kopyasını alıkoyar. Alıkonan yayın, gelecekteki abonelere iletilebilecek şekilde kullanılabilir. Yayıncı uygulaması, yayını bir hedefe seçmede ya da yönlendirmede hiçbir rol oynamaz; görevi, yayın oluşturmak ve yönetici tarafından tanımlanan konulara yayınlar koymak.

Bu sabit konu örneği, birçok yayınlama/abone olunan uygulamanın (atyprice) atomlu bir konudur: statik. Bir sistem yöneticisinin konu dizelerini tanımlamasını ve yayınlandığı konuları değiştirmesini gerektirir. Genel olarak yayınlama/abone olma uygulamalarının bazı ya da tüm konu ağacını tanıması gerekir. Konular

sık sık değişir ya da konular fazla değişmese de, konu bileşimlerinin sayısı büyük ve bir denetimci, yayınlanmasına gerek duyacak her konu dizesi için bir konu düğümü tanımlamada çok zahmetli olur. Belki de konu dizgileri yayınlandığı zaman bilinmez; bir yayıncı uygulaması, bir konu dizgisi belirtmek için yayın içeriğindeki bilgileri kullanabilir ya da bir tarayıcıdan insan girişi gibi başka bir kaynaktan yayınlayabileceğiniz konu dizgileriyle ilgili bilgileri olabilir. Daha dinamik yayınlama biçemleri için bir sonraki örnek, yayınlayıcı uygulamasının bir parçası olarak konuların dinamik olarak nasıl yaratılacağı gösterilir.

Birkaç yayıncı ve aboneyi bir araya getiriniz. Konuları adlandırma ve konu ağaçlarıyla düzenlemek için kuralları ya da mimariyi tasarlama, yayınlama/abone olma çözümünün geliştirilmesinde önemli bir adımdır. Konu ağacının hangi kuruluşun yayınlayıcı ve abone programlarını birbirine bağlayıp birleştirdiği ve bunları konu ağacının içeriğine bağlayacak ölçüde dikkatli bir şekilde bakın. Konu ağacındaki değişikliklerin yayınlayıcıyı ve abone uygulamalarını etkilemesini ve etkiyi nasıl en aza indirebileceğinin sorusunu kendinize sorun. WebSphere MQ yayınlama/abone olma modelinin mimarisine dayalı olarak, bir konunun kök kısmını ya da kök alt ağacını sağlayan bir denetim konusu nesnesi kavramsıdır. Konu nesnesi, uygulama programlama ve işlemlerini basitleştiren ve bunun sonucunda maintainyeteneği iyileştiren, konu ağacı yönetiminin kök kısmını tanımlama seçeneğini size sunar. Örneğin, yalıtılmış konu ağaçlarına sahip birden çok yayınlama/abone olma uygulaması konuşlandırıyorsanız, o zaman konu ağacının kök kısmını yöneterek, farklı uygulamalar tarafından benimsenen konu adlandırma kurallarında tutarlılık olmasa da, konu ağaçlarının yalıtılmasını garanti edebilirsiniz.

Uygulamada, yayınlayıcı uygulamaları, bu örnekte olduğu gibi, yalnızca sabit konuları ve bir sonraki değişken konuları kullanarak bir yelpazeyi kapsamaya devam eder. “Örnek 2: Bir değişken konusuna yayınlayıcı” sayfa 270 , konuların ve konu dizgilerinin kullanımını birleştirmeyi de gösterir.

İlgili kavramlar

“Örnek 2: Bir değişken konusuna yayınlayıcı” sayfa 270
Programsal olarak tanımlanmış bir konuya yayınlama göstermek için bir Websphere MQ programı.

“Abone uygulamaları yazılıyor” sayfa 273
Üç örnek üzerinde çalışarak abone uygulamaları yazmaya başlayın: Bir WebSphere MQ uygulaması, bir kuyruktan iletiler tüketiyor, abonelik yaratan ve kuyruğa alma hakkında bilgi sahibi olmamasını gerektiren bir uygulama ve son olarak hem kuyruklama hem de abonelikler kullanan bir örnek.

Örnek 2: Bir değişken konusuna yayınlayıcı

Programsal olarak tanımlanmış bir konuya yayınlama göstermek için bir Websphere MQ programı.

Not: Sıkıştırılmış kodlama biçemi, üretim kullanımına hazır olmamaya yöneliktir.

See the output in Şekil 40 sayfa 272.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>
int main(int argc, char **argv)
{
    char     topicNameDefault[]   = "STOCKS";
    char     topicStringDefault[] = "IBM/PRICE";
    char     publicationDefault[] = "130";
    MQCHAR48 qmName = "";

    MQHCONN  Hconn = MQHC_UNUSABLE_HCONN; /* connection handle           */
    MQHOBJ   Hobj     = MQHO_NONE;   /* object handle sub queue          */
    MQLONG   CompCode = MQCC_OK;     /* completion code                  */
    MQLONG   Reason   = MQRC_NONE;   /* reason code                      */
    MQOD     td = {MQOD_DEFAULT};    /* Object descriptor                */
    MQMD     md = {MQMD_DEFAULT};    /* Message Descriptor               */
    MQPMO   pmo = {MQPMO_DEFAULT};   /* put message options              */
    MQCHAR   resTopicStr[151];       /* Returned value of topic string */
    char *   topicName   = topicNameDefault;
    char *   topicString = topicStringDefault;
    char *   publication = publicationDefault;
    memset   (resTopicStr, 0 , sizeof(resTopicStr));

    switch(argc){            /* Replace defaults with args if provided */
        default:
            publication = argv[3];
        case(3):
            topicString = argv[2];
        case(2):
            if (strcmp(argv[1],"/"))  /* "/" invalid = No topic object  */
                topicName = argv[1];
            else
                *topicName = '\0';
        case(1):
            printf("Provide parameters:  TopicObject TopicString Publication\n");
    }

    printf("Publish \"%s\" to topic \"%-.48s\" and topic string \"%s\"\n", publication,
topicName, topicString);
    do {
        MQCONN(qmName, &Hconn, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        td.ObjectType = MQOT_TOPIC;    /* Object is a topic             */
        td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4     */
        strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
        td.ObjectString.VSPtr = topicString;
        td.ObjectString.VSLength = (MQLONG)strlen(topicString);
        td.ResObjectString.VSPtr = resTopicStr;
        td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
        MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        pmo.Options = MQPMO_FAIL_IF_QUIESCING | MQPMO_RETAIN;
        MQPUT(Hconn, Hobj, &md, &pmo, (MQLONG)strlen(publication)+1, publication, &CompCode,
&Reason);
        if (CompCode != MQCC_OK) break;
        MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQDISC(&Hconn, &CompCode, &Reason);
    } while (0);
    if (CompCode == MQCC_OK)
        printf("Published \"%s\" to topic string \"%s\"\n",  publication, resTopicStr);
    printf("Completion code %d and Return code %d\n", CompCode, Reason);
}
```

Şekil 39. Yalın WebSphere MQ yayıncısı bir değişken konusuna yayınıdır.

```
X:\Publish2\Debug>PublishStock
Provide parameters:  TopicObject TopicString Publication
Publish "130" to topic "STOCKS" and topic string "IBM/PRICE"
Published "130" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0

X:\Publish2\Debug>PublishStock / NYSE/IBM/PRICE 131
Provide parameters:  TopicObject TopicString Publication
Publish "131" to topic "" and topic string "NYSE/IBM/PRICE"
Published "131" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0
```

Şekil 40. İkinci yayınlayıcı örneğinden alınan örnek çıktı

Bu örneğe dikkat etmek için birkaç nokta vardır.

char topicNameDefault[] = "STOCKS";
Varsayılan konu adı STOCKS , konu dizgisinin bir kısmını tanımlar. Bu adı, programa ilk bağımsız değişken olarak belirterek ya da ilk parametre olarak / belirterek konu adının kullanımını ortadan kaldırarak bu konu adını geçersiz kılabilirsiniz.

char topicString[101] = "IBM/PRICE";
IBM/PRICE , varsayılan konu dizesidir. Bu konu dizilimini, programa ikinci bağımsız değişken olarak belirterek geçersiz kılabilirsiniz.

Kuyruk yöneticisi, STOCKS konu nesnesi ("NYSE") tarafından sağlanan konu dizesini "IBM/PRICE" programı tarafından sağlanan konu dizgisiyle birleştirir ve iki konu dizgisi arasına bir "/" ekler. Sonuç, çözümlenen konu dizgisidir "NYSE/IBM/PRICE". Sonuçta elde edilen konu dizgisi, IBMSTOCKPRICE konu nesnesinde tanımlı olan ve tam olarak aynı etkiye sahip olan dizedir.

Çözülen konu dizgisiyle ilişkilendirilen denetim konusu nesnesinin, yayınlayıcı tarafından MQOPEN ' a iletildiği gibi aynı konu nesnesi olması gerekmez. WebSphere MQ , çözümlenen konu dizgisinde, yayınla ilişkili öznitelikleri hangi denetim konusu nesnesinin tanımladığı şekilde çalışmak için, çözülen konu dizgisinde ağaç örtülü olarak kullanır.

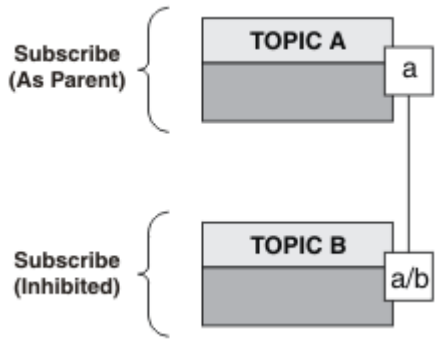
Suppose there are two topic objects A and B, and A defines topic "a", and B defines topic "a/b" (Şekil 41 sayfa 273). If the publisher program refers to topic object A and provides topic string "b", resolving the topic to the topic string "a/b", then the publication inherits its properties from topic object B because the topic matches the topic string "a/b" defined for B.

if (strcmp(argv[1],"/"))
argv[1] , isteğe bağlı olarak sağlanan topicName' dir. "/" bir konu adı olarak geçersiz; burada herhangi bir konu adının olmadığını ve konu dizgisinin tamamen program tarafından sağlandığı anlamına gelir. Şekil 40 sayfa 272 içindeki çıkış, program tarafından devingen olarak sağlanan tüm konu dizesini gösterir.

strncpy(td.ObjectName, topicName, MQ_OBJECT_NAME_LENGTH);
For the default case, the optional topicName needs to be created beforehand, using WebSphere MQ Explorer or this MQSC command:

```
DEFINE TOPIC(STOCKS) TOPICSTR(NYSE) REPLACE;
```

td.ObjectString.VSPtr = topicString;
Konu dizesi, konu tanımlayıcısındaki bir MQCHARV alanıdır.



Şekil 41. Konu Nesne İlişkilendirmeleri

İkinci örnek ne gösteriyor? her ne kadar kod ilk örneğe çok benzer olsa da-etkili bir şekilde sadece iki satır farkı vardır-sonuç, ilk olarak önemli ölçüde farklı bir programdır. Programcılar, yayınların gönderildiği hedeflerin denetlenmesini sağlar. Abone uygulamalarını tasarlamak için kullanılan en düşük yönetici girişiyle birlikte, yayıncılardan abonelere yayınların yönlendirilmesi için hiçbir konu ya da kuyruk önceden tanımlanmaz.

Noktadan noktaya ileti sistemi paradigması içinde, iletilerin akabilmesi için önce kuyruklar tanımlanmalıdır. For publish/subscribe, they do not, although WebSphere MQ implements publish/subscribe using its underlying queuing system; the benefits of guaranteed delivery, transactionality and loose coupling associated with messaging and queueing are inherited by publish/subscribe applications.

Bir tasarımcı yayıncının, abonenin, programların temel konu ağacından haberdar olup olmadığına karar vermek zorundadır. Ayrıca, abone programlarının kuyruktan haberdar olup olmayıp bilmemesi gerekir. Sonraki abonelerin örnek uygulamalarını araştır. Bu örnekler, genellikle `NYSE/IBM/PRICE'` e abone olmak ve abone olmak üzere yayınlayıcı örnekleriyle birlikte kullanılmak üzere tasarlanmıştır.

İlgili kavramlar

"Örnek 1: Sabit bir konuya yayınlayıcı" sayfa 267
Bir yönetimsel olarak tanımlanmış bir konuya yayınlama göstermek için bir WebSphere MQ programı.

"Abone uygulamaları yazılıyor" sayfa 273
Üç örnek üzerinde çalışarak abone uygulamaları yazmaya başlayın: Bir WebSphere MQ uygulaması, bir kuyruktan iletiler tüketiyor, abonelik yaratan ve kuyruğa alma hakkında bilgi sahibi olmamasını gerektiren bir uygulama ve son olarak hem kuyruklama hem de abonelikler kullanan bir örnek.

Abone uygulamaları yazılıyor

Üç örnek üzerinde çalışarak abone uygulamaları yazmaya başlayın: Bir WebSphere MQ uygulaması, bir kuyruktan iletiler tüketiyor, abonelik yaratan ve kuyruğa alma hakkında bilgi sahibi olmamasını gerektiren bir uygulama ve son olarak hem kuyruklama hem de abonelikler kullanan bir örnek.

Çizelge 42 sayfa 274 içinde, tüketicinin ya da abonenin üç stili, bunları karakterize eden WebSphere MQ işlev çağrıları sıralarıyla birlikte listelenir.

1. İlk stil, MQ Publication Consumer (Yayın Tüketicisi), yalnızca MQGET yaptığı MQ programını noktalamak için kullanılan bir noktayla aynıdır. Uygulamanın yayınları tüketmekte olduğu bilgisi yoktur; yalnızca kuyruktan ileti okumaktadır. The subscription that causes publications to get routed to the queue is created administratively using WebSphere MQ Explorer or a command.
2. İkinci biçem, çoğu abone uygulaması için tercih edilen örüntüdür. Abone uygulaması aboneliği oluşturur ve sonra yayınlar alır. Kuyruk yöneticisi, kuyruk yöneticisi tarafından gerçekleştirilir.
3. Üçüncü stilde, abone uygulaması, yayınları doldurmak için kullanılan temel kuyruğu açmayı ve kapatmayı ve kuyruğun yayınlarla doldurulması için abonelikler verir.

One way to understand these styles is to study the example C programs listed in Çizelge 42 sayfa 274 for each of the styles. Örnekler, "Yayınlayıcı uygulamaları yazılıyor" sayfa 267 içinde bulunan yayınlayıcı örneğiyle birlikte çalıştırılacak şekilde tasarlanmıştır.

Çizelge 42. Noktadan noktaya iletişim, WebSphere MQ program örüntüleri ile abone olunması.

Adım	MQ ileti tüketicisi	"Örnek 1: MQ Publication consumer" sayfa 274	"Örnek 2: Yönetilen MQ abonesi" sayfa 276	"Örnek 3: Yönetilmeyen MQ abonesi" sayfa 281
Kuyruk yöneticisine bağlan	MQCONN	MQCONN	MQCONN	MQCONN
Kuyruğu aç	MQOPEN	MQOPEN		MQOPEN
Abone Ol			MQSUB	MQSUB
İleti al	MQGet	MQGet	MQGet	MQGet
Kuyruğu kapat	MQCLOSE	MQCLOSE	(MQCLOSE)	MQCLOSE
Aboneliği kapat			MQCLOSE	MQCLOSE
Kuyruk yöneticisinden bağlantıyı kes	MQDISC	MQDISC	MQDISC	MQDISC

Kaynakları serbest bırakmak, MQCLOSOSE seçeneklerini ya da yalnızca MQOL ile simetri için, MQCLOSOSE ' nin kullanılması her zaman isteğe bağlıdır. Yönetilen MQ abonesi vakasında abonelik kuyruğu kapatıldığında ve simetri bağımsız değişkeni ilgili değilse, MQCLOSE seçeneklerini belirtme olasılığının düşük olduğu için, abonelik kuyruğu Örnek 2: Yönetilen MQ abonesi içinde belirtik olarak kapatılmaz.

Yayınlama/abone olma uygulama kalıplarını anlamanın başka bir yolu da dahil olan farklı varlıklar arasındaki etkileşimlere çok fazla göz atmaktadır. Ömür çizgisi ya da UML sıra çizgeleri etkileşimleri incelemek için iyi bir yoldur. Üç adet ömür çizgisi örneği, "Yaşam çevrimlerini yayınla/abone ol" sayfa 289içinde açıklanır.

Örnek 1: MQ Publication consumer

MQ Publication tüketicisi, konuların kendisine abone olmayan bir IBM WebSphere MQ ileti tükettidir.

To create the subscription and publication queue for this example run the following commands, or define the objects using WebSphere MQ Explorer.

```
DEFINE QLOCAL(STOCKTICKER) REPLACE;
DEFINE SUB(IBMSTOCKPRICESUB) DEST(STOCKTICKER) TOPICOBJ(IBMSTOCKPRICE) REPLACE;
```

The IBMSTOCKPRICESUB subscription references the IBMSTOCK topic object created for the publisher example and the local queue STOCKTICKER. The topic object IBMSTOCK defines the topic string that is used in the subscription, NYSE/IBM/PRICE. Konu nesnesinin ve yayınların alınması için kullanılan kuyruğun, abonelik yaratılmadan önce tanımlanması gerektiğini unutmayın.

MQ yayın tüketici örüntüleri için bir dizi değerli kategori var:

1. Çoklu işlem: Yayınların okuma yazma işlerinden paylaşılması. Yayınlar, abonelik konusuyla ilişkili tek kuyruğa gider. Multiple consumers can open the queue using MQOO_INPUT_SHARED.
2. Merkezi olarak yönetilen abonelikler. Uygulamalar kendi abonelik konularını ya da aboneliklerini oluşturmaz; yayınların gönderildiği yerde yönetici sorumludur.
3. Abonelik yoğunluğu: Birden çok farklı abonelik tek bir kuyruğa gönderilebilir.
4. Abonelik dayanıklılığı: Kuyruğun, tüketicilerin etkin olup olmadığı tüm yayınları alır.
5. Geçiş ve birlikte kullanım: tüketici kodu, bir noktadan noktaya iletişim ve yayınlama/abone olma senaryosu için eşit derecede iyi çalışır.

The subscription creates a relationship between the topic string NYSE/IBM/PRICE and the queue STOCKTICKER. Yürürlükte tutulan yayın da içinde olmak üzere, yayınlar, abonelik yaratıldığı andan itibaren STOCKTICKER ' e iletilir.

Yönetimsel olarak oluşturulan bir abonelik, yönetilebilir ya da yönetilmeyen olabilir. Yönetilen abonelik, yönetilmeyen bir abonelik gibi, yaratıldığı anda yürürlüğe girer. Tüm örüntü kategorileri yönetilen bir abonelik için kullanılabilir değil. Bkz. “Örnek 3: Yönetilmeyen MQ abonesi” sayfa 281

Not: Sıkıştırılmış kodlama biçemi, üretim kullanımına hazır olmamaya yöneliktir.

Sonuçlar Şekil 43 sayfa 276içinde gösterilir.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>
int main(int argc, char **argv)
{
    MQCHAR      publicationBuffer[101];
    MQCHAR48 subscriptionQueueDefault = "STOCKTICKER";
    MQCHAR48 qmName = "";                       /* Use default queue manager    */

    MQHCONN  Hconn = MQHC_UNUSABLE_HCONN;       /* connection handle            */
    MQHOBJ   Hobj = MQHO_NONE;                  /* object handle sub queue      */
    MQLONG   CompCode = MQCC_OK;                /* completion code              */
    MQLONG   Reason = MQRC_NONE;                /* reason code                  */
    MQLONG   messlen = 0;
    MQOD     od = {MQOD_DEFAULT};               /* Unmanaged subscription queue */
    MQMD     md = {MQMD_DEFAULT};               /* Message Descriptor           */
    MQGMO    gmo = {MQGMO_DEFAULT};             /* Get message options          */
    char *   publication=publicationBuffer;
    char *   subscriptionQueue = subscriptionQueueDefault;

    switch(argc){                     /* Replace defaults with args if provided */
        default:
            subscriptionQueue = argv[1]
        case(1):
            printf("Optional parameter: subscriptionQueue\n");
    }

    do {
        MQCONN(qmName, &Hconn, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        strncpy(od.ObjectName, subscriptionQueue, MQ_Q_NAME_LENGTH);
        MQOPEN(Hconn, &od, MQOO_INPUT_AS_Q_DEF | MQOO_FAIL_IF_QUIESCING , &Hobj, &CompCode,
&Reason);
        if (CompCode != MQCC_OK) break;
        gmo.Options = MQGMO_WAIT | MQGMO_NO_SYNCPOINT | MQGMO_CONVERT;
        gmo.WaitInterval = 10000;
        printf("Waiting %d seconds for publications from %s\n", gmo.WaitInterval/1000,
subscriptionQueue);
        do {
            memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
            memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
            md.Encoding       = MQENC_NATIVE;
            md.CodedCharSetId = MQCCSI_Q_MGR;
            memset(publication, 0, sizeof(publicationBuffer));
            MQGET(Hconn, Hobj, &md, &gmo, sizeof(publicationBuffer)-1, publication, &messlen,
                  &CompCode, &Reason);
            if (Reason == MQRC_NONE)
                printf("Received publication \"%s\"\n", publication);
        }
        while (CompCode == MQCC_OK);
        if (CompCode != MQCC_OK && Reason != MQRC_NO_MSG_AVAILABLE) break;
        MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQDISC(&Hconn, &CompCode, &Reason);
    } while (0);
    printf("Completion code %d and Return code %d\n", CompCode, Reason);
}
```

Şekil 42. MQ yayın tüketicisi.

```
X:\Subscribe1\Debug>Subscribe1
Optional parameter: subscriptionQueue
Waiting 10 seconds for publications from STOCKTICKER
Received publication "129"
Completion code 0 and Return code 0
```

Şekil 43. MQ yayın tüketicisinin çıkışı

Aşağıdakilerden haberdar olmak için bir çift standart WebSphere MQ *C* dil programlama ipucu vardır:

memset(publication, 0, sizeof(publicationBuffer));
Ensure the message has a trailing null for easy formatting using `printf`. Yayınlayıcı örneği, 1-`strlen(publication)` eklenerek `MQPUT` ' a geçirilen ileti arabelleğindeki sondaki boş değeri içerir. `MQCHAR` arabelleklerinin boş değere ayarlanması, dizgileri saklamak için arabellekleri kullanan IBM WebSphere MQ C programları için iyi bir programlama stilidir ve boş değer, arabelleği tam olarak dolduramayan bir karakter dizisini izlemektedir.

MQGET(Hconn, Hobj, &md, &gmo, sizeof(publicationBuffer)-1, publication, &messlen, &CompCode, &Reason);
Döndürülebilmek için, ileti arabelleğinin sonunda bir boş (null) değeri, "`if (messlen == strlen(publication));`" ' un true (doğru) olması durumunda döndürülen iletinin boş değerli olduğunu doğrulayın. Bu ipucu, önceki bir ipucunu tamamlar ve `publicationBuffer` ' ta, `publication`içeriğinin üzerine yazılmamış en az bir boş değer olmasını sağlar.

İlgili kavramlar

“Örnek 2: Yönetilen MQ abonesi” sayfa 276
Yönetilen MQ abonesi, çoğu abone uygulaması için tercih edilen kalıbdır. Örneğin, kuyrukların *hayır* yönetim tanımlaması, konuları ya da abonelikleri gerektirir.

“Örnek 3: Yönetilmeyen MQ abonesi” sayfa 281
Yönetilmeyen abone, önemli bir abone uygulaması sınıfıdır. Bununla birlikte, yayınların ve yayınların tüketilmesinin *denetimi* ile yayınlama/abone olma avantajlarını birleştirin. Bu örnek, abonelikleri ve kuyrukları birleştirmenin farklı yollarını gösterir.

“Yayınlayıcı uygulamaları yazılıyor” sayfa 267
İki örnek üzerinde çalışarak yayınlayıcı uygulamaları yazmaya başlayın. İlki, bir kuyruğa ileti yerleştirmek için bir noktadan noktaya mümkün olduğunca yakın bir şekilde modellenir ve ikinci olarak konular, yayınlayıcı uygulamaları için dinamik olarak daha yaygın bir kalıba yol açmaktadır.

Örnek 2: Yönetilen MQ abonesi

Yönetilen MQ abonesi, çoğu abone uygulaması için tercih edilen kalıbdır. Örneğin, kuyrukların *hayır* yönetim tanımlaması, konuları ya da abonelikleri gerektirir.

This simplest kind of managed subscriber typically uses a *dayanıklı olmayan* subscription. Örnek, dayanıklı olmayan bir aboneliğe odaklanır. The subscription lasts only as long as the lifetime of the subscription handle from `MQSUB`. Any publications that match the topic string during the lifetime of the subscription are sent to the subscription queue (and possibly a retained publication if the flag `MQSO_NEW_PUBLICATIONS_ONLY` is not set or defaulted, an earlier publication matching the topic string was retained, and the publication was persistent or the queue manager has not terminated, since the publication was created).

Ayrıca, bu kalıpla *dayanıklı* bir abonelik de kullanabilirsiniz. Typically if a managed durable subscription is used it is done for reliability reasons, rather than to establish a subscription that, without any errors occurring, would outlive the subscriber. For more information about different lifecycles associated with managed, unmanaged, durable and non-durable subscriptions see the related topics section.

Sürekli abonelikler genellikle kalıcı yayınlarla ve kalıcı olmayan yayınlarla, kalıcı olmayan aboneliklerle ilişkilendirilir, ancak abonelik dayanıklılığı ile yayın sürekliliği arasında gerekli bir ilişki yoktur. Tüm kalıcılık ve dayanıklılık birleşimleri mümkündür.

Yönetilen kalıcı olmayan vaka dikkate alındığında, kuyruk yöneticisi, kuyruk kapatıldığında temizlenen ve silinen bir abonelik kuyruğu oluşturur. Bu yayınlar, kalıcı olmayan abonelik kapatıldığında kuyruktan kaldırılır.

Bu kod tarafından belirtilen yönetilen dayanıklı olmayan örüntüye ilişkin değerli iç işlev kümeleri aşağıdaki gibidir:

1. talep aboneliğine: abonelik konu dizgisi dinamiktir. Uygulama çalıştırıldığında uygulama tarafından sağlanır.
2. Kendi kendini yöneten kuyruk: Abonelik kuyruğu kendi kendini tanımlıyor ve yönetmekte.
3. Otomatik olarak abonelik yaşam çevrimini yönetme: *non-durable* abonelikleri yalnızca abone uygulamasının süresi için var olur.
   - Bir *dayanıklı* yönetilen abonelik tanımlarsaniz, bu durumda kalıcı bir abonelik kuyruğu ve yayınların etkinolan abone programları olmadan saklanabilmesiyle sonuçlanır. Kuyruk yöneticisi, yalnızca uygulama ya da yönetici aboneliği silmeyi seçtikten sonra, kuyruğu siler (ve alınmamış yayınları bundan temizler). The subscription can be deleted using an administrative command, or by closing the subscription with the `MQCO_REMOVE_SUB` option.
   - Consider setting `SubExpiry` for durable subscriptions so that publications cease to be sent to the queue and the subscriber can consume any remaining publications before removing the subscription and causing the queue manager to delete the queue and any remaining publications on it.
4. Esnek konu dizesi devreye alımı: Abonelik konu yönetimi, yönetimsel olarak tanımlanmış bir konu kullanılarak aboneliğin kök kısmı tanımlanarak basitleştirilmiştir. Daha sonra, konu ağacının kök kısmı uygulamadan gizlenir. By hiding the root part an application can be deployed without the application inadvertently creating a topic tree that overlaps with another topic tree created by another instance, or another application.
5. Administered topics: by using a topic string in which the first part matches an administratively defined topic object, publications are managed according to the attributes of the topic object.
   - örneği için, konu dizgisinin ilk bölümü, kümelenmiş bir konu nesnesiyle ilişkilendirilmiş konu dizgisiyle eşleşiyorsa, abonelik, kümenin diğer üyelerinden yayınları alabilir.
   - Yönetimsel olarak tanımlanmış konu nesnelerine ve programsal olarak tanımlanmış aboneliklere ilişkin seçici eşleştirme, her ikisinin de avantajlarını birleştirmenizi sağlar. Sistem yöneticisi konular için öznitelikler sağlar ve programcılar, konuların yönetimiyle ilgilenmeden, "alt-topicskonularını" tanımlar.
   - It is the resultant topic string which is used to match the topic object that provides the attributes associated with the topic, and not necessarily the topic object named in `sd.Objectname`, although they typically turn out to be one and the same. Bkz. “Örnek 2: Bir değişken konusuna yayınlayıcı” sayfa 270.

By making the subscription durable in the example, publications continue to be sent to the subscription queue after the subscriber has closed the subscription with the `MQCO_KEEP_SUB` option. Kuyruk etkin olmadığında, kuyruk yayınları almaya devam eder. You can override this behavior by creating the subscription with the `MQSO_PUBLICATIONS_ON_REQUEST` option and using `MQSUBRQ` to request the retained publication.

Abonelik, daha sonra `MQCO_RESUME` seçeneği ile birlikte abonelik açılarak sürdürülür.

You can use the queue handle, `Hobj`, returned by `MQSUB` in a number of ways. Kuyruk tanıtıcısı, örnek olarak, abonelik kuyruğunun adını sorgulamak için kullanılır. Yönetilen kuyruklar, `SYSTEM.NDURABLE.MODEL.QUEUE` ya da `SYSTEM.DURABLE.MODEL.QUEUE`varsayılan model kuyrukları kullanılarak açılmıştır. You can override the defaults by providing your own durable and non-durable model queues on a topic by topic basis as properties of the topic object associated with the subscription.

Model kuyruklarından devralınan özniteliklerden bağımsız olarak, ek abonelik yaratmak için yönetilen bir kuyruk tanıtıcısını yeniden kullanamazsınız. Ayrıca, yönetilen kuyruğu, döndürülen kuyruk adını kullanarak ikinci kez açarak, yönetilen kuyruk için başka bir tanıtıcı elde edebilirdiniz. Kuyruk, dışlayıcı girişiiçin açılmış gibi işlev görür.

Yönetilmeyen kuyruklar, yönetilen kuyruklardan daha esnekdir. Örneğin, yönetilmeyen kuyrukları paylaşabilir ya da bir kuyrukta birden çok abonelik tanımlayabilirsiniz. Sonraki örnek, “Örnek 3: Yönetilmeyen MQ abonesi” sayfa 281, aboneliklerin yönetilmeyen bir abonelik kuyruğuyla nasıl birleştirileceğini gösterir.

Not: Sıkıştırılmış kodlama biçemi, üretim kullanımına hazır olmamaya yöneliktir.

Sonuçlar Şekil 46 sayfa 279içinde gösterilir.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>

void inquireQname(MQHCONN HConn, MQHOBJ Hobj, MQCHAR48 qName);

int main(int argc, char **argv)
{
    MQCHAR48 topicNameDefault     = "STOCKS";
    char     topicStringDefault[] = "IBM/PRICE";
    MQCHAR48 qmName = "";                 /* Use default queue manager     */
    MQCHAR48 qName = "";                  /* Allocate to query queue name  */
    char     publicationBuffer[101];      /* Allocate to receive messages  */
    char     resTopicStrBuffer[151];   /* Allocate to resolve topic string */

    MQHCONN  Hconn = MQHC_UNUSABLE_HCONN; /* connection handle             */
    MQHOBJ   Hobj = MQHO_NONE;            /* publication queue handle      */
    MQHOBJ   Hsub = MQSO_NONE;            /* subscription handle           */
    MQLONG   CompCode = MQCC_OK;          /* completion code               */
    MQLONG   Reason = MQRC_NONE;          /* reason code                   */
    MQLONG   messlen = 0;
    MQSD     sd = {MQSD_DEFAULT};         /* Subscription Descriptor       */
    MQMD     md = {MQMD_DEFAULT};         /* Message Descriptor            */
    MQGMO   gmo = {MQGMO_DEFAULT};        /* get message options           */

    char *   topicName   = topicNameDefault;
    char *   topicString = topicStringDefault;
    char *   publication = publicationBuffer;
    char *   resTopicStr = resTopicStrBuffer;
    memset(resTopicStr, 0, sizeof(resTopicStrBuffer));

    switch(argc){               /* Replace defaults with args if provided */
        default:
          topicString = argv[2];
        case(2):
          if (strcmp(argv[1],"/"))       /* "/" invalid = No topic object  */
             topicName = argv[1];
          else
             *topicName = '\0';
        case(1):
             printf("Optional parameters: topicName, topicString\nValues \"%s\"  \"%s\"\n",
                   topicName, topicString);
    }
```

Şekil 44. Yönetilen MQ abonesi-bölüm 1: bildirimler ve parametre işleme.

Bu örnekteki bildirimlere ilişkin olarak bazı ek açıklamalar da vardır.

MQHOBJ Hobj = MQHO_NONE;
You cannot explicitly open a non-durable managed subscription queue to receive publications, but you do need to allocate storage for the object handle the queue manager returns when it opens the queue for you. Bu tanıtıcıyı MQHO_OBJECTolarak kullanıma hazırlamak önemlidir. Bu , kuyruk yöneticisine, abonelik kuyruğuna bir kuyruk tanıtıcısı döndürmesi gerektiğini belirtir.

MQSD sd = {MQSD_DEFAULT};
MQSUBiçinde kullanılan yeni abonelik tanımlayıcısı.

MQCHAR48 qName;
Although the example doesn't require knowledge of the subscription queue, the example does inquire the name of the subscription queue - the MQINQ binding is a little awkward in the C language, so you might find this part of the example useful to study.

```
    do {
        MQCONN(qmName, &Hconn, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        strncpy(sd.ObjectName, topicName, MQ_TOPIC_NAME_LENGTH);
        sd.ObjectString.VSPtr = topicString;
        sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
        sd.Options = MQSO_CREATE | MQSO_MANAGED | MQSO_NON_DURABLE | MQSO_FAIL_IF_QUIESCING ;
        sd.ResObjectString.VSPtr = resTopicStr;
        sd.ResObjectString.VSBufSize = sizeof(resTopicStrBuffer)-1;
        MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        gmo.Options = MQGMO_WAIT | MQGMO_NO_SYNCPOINT | MQGMO_CONVERT;
        gmo.WaitInterval = 10000;
        inquireQname(Hconn, Hobj, qName);
        printf("Waiting %d seconds for publications matching \"%s\" from \"%-0.48s\"\n",
               gmo.WaitInterval/1000, resTopicStr, qName);
        do {
            memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
            memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
            md.Encoding       = MQENC_NATIVE;
            md.CodedCharSetId = MQCCSI_Q_MGR;
            memset(publicationBuffer, 0, sizeof(publicationBuffer));
            MQGET(Hconn, Hobj, &md, &gmo, sizeof(publicationBuffer-1),
                publication, &messlen, &CompCode, &Reason);
            if (Reason == MQRC_NONE)
                printf("Received publication \"%s\"\n", publication);
        }
        while (CompCode == MQCC_OK);
        if (CompCode != MQCC_OK && Reason != MQRC_NO_MSG_AVAILABLE) break;
        MQCLOSE(Hconn, &Hsub, MQCO_REMOVE_SUB, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQDISC(&Hconn, &CompCode, &Reason);
    } while (0);
    printf("Completion code %d and Return code %d\n", CompCode, Reason);
    return;
}
void inquireQname(MQHCONN Hconn, MQHOBJ Hobj, MQCHAR48 qName) {
#define _selectors 1
#define _intAttrs 1

    MQLONG select[_selectors] = {MQCA_Q_NAME}; /* Array of attribute selectors */
    MQLONG intAttrs[_intAttrs];                /* Array of integer attributes  */
    MQLONG CompCode, Reason;
    MQINQ(Hconn, Hobj, _selectors, select, _intAttrs, intAttrs, MQ_Q_NAME_LENGTH, qName,
          &CompCode, &Reason);
    if (CompCode != MQCC_OK) {
        printf("MQINQ failed with Condition code %d and Reason %d\n", CompCode, Reason);
        strcpy(qName, "unknown queue");
    }
    return;
}
```

Şekil 45. Yönetilen MQ abonesi-bölüm 2: kod gövdesi.

```
W:\Subscribe2\Debug>solution2
Optional parameters: topicName, topicString
Values "STOCKS"  "IBM/PRICE"
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from
"SYSTEM.MANAGED.NDURABLE.48A0AC7403300020        "
Received publication "150"
Completion code 0 and Return code 0

W:\Subscribe2\Debug>solution2 / NYSE/IBM/PRICE
Optional parameters: topicName, topicString
Values ""  "NYSE/IBM/PRICE"
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from
"SYSTEM.MANAGED.NDURABLE.48A0AC7403310020        "
Received publication "150"
Completion code 0 and Return code 0
```

Şekil 46. Yönetilen MQ abonesinden çıkış

Bu örnekteki kod hakkında yapılacak ek açıklamalar da vardır.

strncpy(sd.ObjectName, topicName, MQ_Q_NAME_LENGTH);
topicName boş değerli ya da boş *(varsayılan değer)*ise, çözümlenen konu dizesini hesaplamak için konu adı kullanılmaz.

sd.ObjectString.VSPtr = topicString;
Önceden tanımlanmış bir konu nesnesini kullanmak yerine, bu örnekte programcı, MQSUBile birleştirilen bir konu nesnesi ve bir konu dizgisi sağlar. Konu dizgisinin bir MQCHARV yapısı olduğunu fark edin.

sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
Bir MQCHARV alanının uzunluğunu ayarlamaya alternatif bir seçenek.

sd.Options = MQSO_CREATE | MQSO_MANAGED | MQSO_NON_DURABLE | MQSO_FAIL_IF_QUIESCING;
Konu dizesini tanımladıktan sonra, sd.Options işaretlerinin en dikkat çekmesine gerek vardır. There are many options, the example specifies only the most commonly used ones; the others are left to default.

1. As the subscription is *dayanıklı olmayan*, that is, it has a lifetime of the open subscription in the application, set the MQSO_CREATE flag. Ayrıca okunabilirlik için de *(varsayılan)* MQSO_NON_DURABLE işaretini ayarlayabilirsiniz.
2. Complemening MQSO_CREATE , MQSO_RESUME. Both flags can be set together; the queue manager either creates a new subscription or resumes an existing subscription, whichever is appropriate. However, if you do specify MQSO_RESUME you must also initialize the MQCHARV structure for sd.SubName, even if there is no subscription to resume. Failure to initialize SubName results in a return code of 2440: MQRC_SUB_NAME_ERROR from MQSUB.

   Not: MQSO_RESUME is always ignored for a non-durable managed subscription: but specifying it without initializing the MQCHARV structure for sd.SubName does cause the error.
3. Buna ek olarak, aboneliğin nasıl açılacağını etkileyen üçüncü bir işaret de vardır, MQSO_ALTER. Doğru izinleri göz önüne alındığında, devam ettiren bir aboneliğin özellikleri, MQSUB' ta belirtilen diğer özniteliklerle eşleşecek şekilde değiştirilir.

   Not: MQSO_CREATE, MQSO_RESUME ve MQSO_ALTER işaretlerinden en az birinin belirtilmesi gerekir. Bkz. Seçenekler (MQUZE). "Örnek 3: Yönetilmeyen MQ abonesi" sayfa 281' ta üç işaretin tümünü kullanmanın örnekleri vardır.
4. Otomatik olarak sizin için aboneliği yönetmek üzere kuyruk yöneticisi için MQSO_MANAGED ' yi ayarlayın.

sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
İsteğe bağlı olarak, boş sonlandırılmış dizgiler için MQCHARV uzunluğunun ayarını kaldırın ve bunun yerine boş değerli sonlandırıcı işaretini kullanın.

sd.ResObjectString.VSPtr = resTopicStr;
Sonuçtaki konu dizgisi, programdaki ilk printf içinde yankılanır. Çözümlenen dizgiyi programa geri döndürmek için MQCHARV ResObjectString for WebSphere MQ ' yı ayarlayın.

Not: resTopicStringBuffer , memset(resTopicStr, 0, sizeof(resTopicStrBuffer))' ta boş değerler (null) olarak kullanıma hazırlandı. Döndürülen konu dizgileri sondaki boş değerle bitmiyor.

sd.ResObjectString.VSBufSize = sizeof(resTopicStrBuffer)-1;
sd.ResObjectString ' ın arabellek büyüklüğünü, gerçek büyüklüğünden daha az bir değerine ayarlayın. Bu , çözümlenen konu dizesinin tüm arabelleği dolduracağı durumlarda, sağlanan boş değerli sonlandırıcının üzerine yazılmasını önler.

Not: Konu dizgisi sizeof(resTopicStrBuffer)-1' den uzunsa, hata döndürülmez. Even if VSLength > VSBufSiz the length returned in sd.ResObjectString.VSLength is the length of the complete string and not necessarily the length of the returned string. Konu dizgisinin tamamlandığını doğrulamak için sd.ResObjectString.VSLength < sd.ResObjectString.VSBufSiz sınamasını test edin.

MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
MQSUB işlevi bir abonelik yaratır. If it is non-durable you are probably not interested in its name, though you can inspect its status in WebSphere MQ Explorer. `sd.SubName` parametresini girdisiolarak sağlayabilirsiniz, bu nedenle hangi adı arayabileceğinin farkında olabilirsiniz; açıkça, diğer aboneliklerle ad çakışmalarını önlemeniz gerekir.

MQCLOSE(Hconn, &Hsub, MQCO_REMOVE_SUB, &CompCode, &Reason);
Hem abonelik, hem de abonelik kuyruğu kapatılıyor. Örnekte, abonelik kapatılır, ancak kuyruk kapatılmaz. Abonelik, kalıcı olmayan bir abonelik olduğunda, bu durumda varsayılan olarak `MQCLOSE` `MQCO_REMOVE_SUB` seçeneği varsayılan değerdir. `MQCO_KEEP_SUB` kullanımı bir hatadır.

Not: the subscription *kuyruk* is not closed by `MQSUB`, and its handle, `Hobj`, remains valid until the queue is closed by `MQCLOSE` or `MQDISC`. Uygulama zamanından önce sona erdirilirse, kuyruk ve abonelik, uygulama sona erdirildikten sonra kuyruk yöneticisi tarafından temizlenir.

İlgili kavramlar
"Örnek 1: MQ Publication consumer" sayfa 274
MQ Publication tüketicisi, konuların kendisine abone olmayan bir IBM WebSphere MQ ileti tükettidir.

"Örnek 3: Yönetilmeyen MQ abonesi" sayfa 281
Yönetilmeyen abone, önemli bir abone uygulaması sınıfıdır. Bununla birlikte, yayınların ve yayınların tüketilmesinin *denetimi* ile yayınlama/abone olma avantajlarını birleştirin. Bu örnek, abonelikleri ve kuyrukları birleştirmenin farklı yollarını gösterir.

"Yayınlayıcı uygulamaları yazılıyor" sayfa 267
İki örnek üzerinde çalışarak yayınlayıcı uygulamaları yazmaya başlayın. İlki, bir kuyruğa ileti yerleştirmek için bir noktadan noktaya mümkün olduğunca yakın bir şekilde modellenir ve ikinci olarak konular, yayınlayıcı uygulamaları için dinamik olarak daha yaygın bir kalıba yol açmaktadır.

Örnek 3: Yönetilmeyen MQ abonesi
Yönetilmeyen abone, önemli bir abone uygulaması sınıfıdır. Bununla birlikte, yayınların ve yayınların tüketilmesinin *denetimi* ile yayınlama/abone olma avantajlarını birleştirin. Bu örnek, abonelikleri ve kuyrukları birleştirmenin farklı yollarını gösterir.

The unmanaged pattern is more commonly associated with *dayanıklı* subscriptions than *dayanıklı olmayan*. Tipik olarak, yönetilmeyen bir abone tarafından yaratılan bir aboneliğin yaşam çevrimi, abone olan uygulamanın yaşam çevriminden bağımsızdır. Abonelik dayanıklı hale getirilerek, abonelik etkin olmadığında da abonelik alır.

Aynı sonucu elde etmek için dayanıklı *yönetilen* abonelikler oluşturabilirsiniz, ancak bazı uygulamalar, yönetilen abonelikle mümkün olan kuyruklar ve iletiler üzerinde daha fazla esneklik ve denetim gerektirir. Kalıcı olarak yönetilen bir abonelik için, kuyruk yöneticisi, abonelik konularıyla eşleşen yayınlar için kalıcı bir kuyruk yaratır. Abonelik silindiğinde kuyruğun ve ilişkili yayınların silinmesine neden olur.

Genellikle dayanıklı *yönetilen* abonelikler, uygulamanın yaşam çevrimi ve abonelik temelde aynıysa, ancak garanti vermek için çok zorsa kullanılır. Abonelik kalıcı hale getirerek ve yayınlayıcının kalıcı yayınlar yaratması nedeniyle, kuyruk yöneticisi ya da abonenin zamanından önce sona ermesi ve kurtarılması gerekirken, kayıp ileti bulunamaması gerekir.

Kuyruk yöneticisi, bir abone için kalıcı olarak yönetilen abonelik kuyruğunu örtük olarak açar; böyle bir yolla, kuyruğun işlenmesi olanaklı değildir. Buna ek olarak, her yönetilen kuyruk için birden çok abonelik yaratamazsınız ve kuyrukların adları üzerinde daha az denetime sahip olduğunuz için kuyrukları yönetmek için daha zor bulabilirsiniz. Bu nedenlerden dolayı, *yönetilmeyen* MQ abonesinin, *yönetilen* MQ abonesinden dayanıklı abonelikler gerektiren uygulamalar için daha uygun olup olmadığını göz önünde bulundurun.

Şekil 49 sayfa 286 içindeki kod, yönetilmeyen bir kalıcı abonelik örüntmesini gösterir. Şekil için, kodun yönetilmeyen, kalıcı olmayan abonelikleri de yaratılmasına neden olur. Bu örnek, aşağıdaki örüntü kategorilerini gösterir:

- İstek üzerine abonelikler: abonelik konu dizgileri dinamiktir. Bunlar, uygulama çalıştırıldığında uygulama tarafından sağlanır.

- Basitleştirilmiş abonelik konu yönetimi: abonelik konusu yönetimi, bir yönetimsel olarak tanımlanmış bir konu kullanılarak abonelik konu dizgisinin kök kısmı tanımlanarak basitleştirilmiştir. Bu, uygulamadaki konu ağacının kök kısmını gizler. Bir abonenin kök kısmını gizleyerek farklı konu ağaçlarına konuşlandırılabilir.
- Esnek abonelik yönetimi: Bir aboneliği yönetimsel olarak tanımlayabilir ya da bir abone programında isteğe bağlı olarak yaratabilirsiniz. Aboneliğin nasıl oluşturulduğunu gösteren bir öznitelik dışında, yönetimsel olarak programlı olarak abonelikler arasında bir fark yoktur. Aboneliklerin dağıtılması için kuyruk yöneticisi tarafından otomatik olarak oluşturulan üçüncü bir abonelik tipi vardır. Tüm abonelikler WebSphere MQ Explorer 'da görüntülenir.
- Kuyruklara sahip aboneliklerin esnek ilişkilendirmesi: Önceden tanımlanmış bir yerel kuyruk, MQSUB işlevi tarafından bir abonelikle ilişkilendirilir. Abonelikleri kuyruklarla ilişkilendirmek için MQSUB ' yi kullanmanın farklı yolları vardır:
  - Var olan abonelikleri *yok* olan bir kuyrukla ilişkilendirmeyi ilişkilendirin `MQSO_CREATE + (Hobj from MQOPEN)`.
  - Bir *yeni* aboneliğini, var olan aboneliklere sahip bir kuyrukla ilişkilendirin, `MQSO_CREATE + (Hobj from MQOPEN)`.
  - Var olan bir aboneliğin farklı bir kuyruğa taşınması, `MQSO_ALTER + (Hobj from MQOPEN)`.
  - Resume an existing subscription associated with an existing queue, `MQSO_RESUME + (Hobj = MQHO_NONE)`, or `MQSO_RESUME + (Hobj = from MQOPEN of queue with existing subscription)` .
  - By combining `MQSO_CREATE | MQSO_RESUME | MQSO_ALTER` in different combinations, you can cater for different input states of the subscription and the queue without having to code multiple versions of MQSUB with different `sd.Options` values.
  - Alternatively, by coding a specific choice of `MQSO_CREATE | MQSO_RESUME | MQSO_ALTER` the queue manager returns an error (Çizelge 43 sayfa 283) if the states of the subscription and queue provided as input to MQSUB are inconsistent with the value of `sd.Options`. Şekil 55 sayfa 288 shows the results of issuing MQSUB for Subscription X with different individual settings of the `sd.Options` flag, and passing it three different object handles.

  Bu farklı tür hatalara bilgi sahibi olmak için Şekil 48 sayfa 285 içindeki örnek programa farklı girişleri keşfedin. Çizelgede listelenen vakalara dahil olmayan `RC = 2440`ortak bir hata, bir abonelik adı hatasıdır. `MQSO_RESUME` ya da `MQSO_ALTER` ile boş değerli ya da geçersiz bir abonelik adı geçirilerek bu neden kaynaklanır.
- Çoklu işlem: Yayınların okuması için birçok tüketicinin arasında paylaşıma sahip olabilirsiniz. Yayınlar, abonelik konusuyla ilişkili tek kuyruğa gider. Consumers have a choice of opening the queue directly using MQOPEN or resuming the subscription using MQSUB.
- Abonelik yoğunluğu: Aynı kuyruğun birden çok aboneliği yaratılabilir. "Çakışan" aboneliklere yol açabileceği ve aynı yayını birden çok kez alan bu yetenek için dikkatli olun. `MQSO_GROUP_SUB` seçeneği, çakışan aboneliklerin neden olduğu yinelenen yayınları ortadan kaldırır.
- abone ve tüketici ayrımı: örneklerde resimli üç tüketici modeli ile bir diğer model de tüketiciyi aboneden ayırmak. Yönetilmeyen MQ Abonesi 'nin bir varyasyonu, aynı programda MQOPEN ve MQSUB , yayınlara abone olan bir program ve başka bir program bunları tüketir. Örneğin, abone bir yayınlama/abone olma kümesinin bir parçası olabilir ve kuyruk yöneticisi kümesi dışında bir kuyruk yöneticisine bağlı tüketici olabilir. Tüketici, abonelik kuyruğunu uzak kuyruk tanımlaması olarak tanımlayarak standart dağıtılmış kuyruklama yoluyla yayınlar alır.

Özellikle de bu seçeneklerin birleşimlerini kullanarak kodunuzu basitleştirmeyi planlıyorsanız, `MQSO_CREATE | MQSO_RESUME | MQSO_ALTER` ' un davranışını anlamak önemlidir. Study the table Çizelge 43 sayfa 283 that shows the results of passing different queue handles to MQSUB, and the results of running the example program shown in Şekil 50 sayfa 287 to Şekil 55 sayfa 288.

Tabloyu oluşturmak için kullanılan senaryoda bir abonelik X ve iki kuyruk, A ve B bulunur. `sd.SubName` abonelik adı parametresi Xolarak ayarlandı, Akuyruğuna eklenmiş bir aboneliğin adı. B kuyruğu, bu kuyruğa bağlı bir aboneliğe sahip değil.

In Çizelge 43 sayfa 283, MQSUB is passed subscription X and the queue handle to queue A. Abonelik seçenekleri ile ilgili sonuçlar aşağıdaki gibidir:

- Kuyruk tanıtıcısı, zaten Xaboneliği olan A kuyruğuna karşılık geldiği içinMQSO_CREATE başarısız oluyor. Başarılı çağrıya karşılık bu davranışı karşılaştırın. That call succeeds because queue B does not have a subscription to X attached to it.
- MQSO_RESUME succeeds because the queue handle corresponds to the queue A which already has a subscription to X. In contrast, the call fails where the subscription X does not exist on queue A.
- MQSO_ALTER , abonelik ve kuyruk açılmasına ilişkin olarak MQSO_RESUME ' e benzer bir şekilde hareket eder. However if the attributes contained within the subscription descriptor passed to MQSUB differ from the attributes of the subscription, MQSO_RESUME fails, whereas MQSO_ALTER succeeds as long as the program instance has permission to alter the attributes. Bir abonelikte konu dizesini hiçbir zaman değiştiremeyebileceğiniz, ancak bir hata döndürmektense, MQSUB , abonelik tanımlayıcısındaki konu adı ve konu dizgisi değerlerini yoksayar ve var olan abonelikte değerleri kullanır.

Daha sonra, Çizelge 43 sayfa 283 'a bakın; burada MQSUB, XX aboneliği geçirilir ve kuyruk B' ye ilişkin kuyruk sapmasını içerir. Abonelik seçenekleri ile ilgili sonuçlar aşağıdaki gibidir:

- MQSO_CREATE succeeds and creates subscription X on queue B because this is a new subscription on queue B.
- MQSO_RESUME başarısız olur. MQSUB , B kuyruğunda X aboneliği olup olmadığını arar ve onu bulmaz, ancak *RC = 2428-subscription X 'in yok* döndürülmesi yerine, *RC = 2019-Abonelik kuyruğu kuyruk nesne tanıtıcısı ile eşleşmez*değerini döndürür. MQSO_ALTER üçüncü seçeneğindeki davranış, bu beklenmeyen hatanın nedenini belirtir. MQSUB , kuyruk sapının bir aboneliği olan bir kuyruğu göstermesini bekler. sd.SubName içinde belirtilen aboneliğin var olup olmadığını denetlemeden önce bunu denetler.
- MQSO_ALTER succeeds, and moves the subscription from queue A to queue B.

A case that is not shown in the table is if the subscription name of the subscription on queue A does not match the subscription name in sd.SubName. Bu arama, *RC = 2428-Kuyruk A ' da abonelik X yok* ile başarısız olur.

Çizelge 43. Farklı kuyruk tanıtıcıları ve abonelik birleşimleriyle MQSUB ' daki hatalar

	Kuyruk A **Abonelik X** **Kuyruk B** **Abonelik yok**	**Kuyruk A** **Abonelik yok** **Kuyruk B** **Abonelik yok**
Hobj for *Kuyruk A* , MQSUB ' ye geçti	**MQSO_CREATE** RC = 2432-Abonelik X, Kuyruk A ' da zaten var **MQSO_RESUME** Kuyruk A üzerindeki X aboneliğini sürdürür **MQSO_ALTER** Kuyruk A ' daki X aboneliğini sürdürür ve izin verilen değişiklikleri yapar	**MQSO_CREATE** Kuyruk A ' da abonelik X 'i yaratır **MQSO_RESUME** RC = 2428-Kuyruk A ' da Abonelik X yok **MQSO_ALTER** RC = 2428-Kuyruk A ' da Abonelik X yok
Hobj for *Kuyruk B* , MQSUB ' ye geçti	**MQSO_CREATE** Kuyruk B ' de yeni abonelik X yaratır **MQSO_RESUME** RC = 2019-Abonelik kuyruğu, kuyruk nesnesi tanıtıcısı ile eşleşmiyor **MQSO_ALTER** X aboneliği X 'i Kuyruk A 'dan Kuyruk B' ye taşı	**MQSO_CREATE** Kuyruk B ' de yeni abonelik X yaratır **MQSO_RESUME** RC = 2428-Kuyruk B ' de abonelik X yok **MQSO_ALTER** RC = 2428-Kuyruk B ' de abonelik X yok

Çizelge 43. Farklı kuyruk tanıtıcıları ve abonelik birleşimleriyle MQSUB ' daki hatalar (devamı var)		
	Kuyruk A **Abonelik X** **Kuyruk B** **Abonelik yok**	**Kuyruk A** **Abonelik yok** **Kuyruk B** **Abonelik yok**
MQHO_NONE, MQSUB ' a geçti	**MQSO_CREATE** RC = 2019-Hatalı nesne tanıtıcısı: yönetilen abonelik yaratmak ve yönetilen bir kuyruk yaratmak için MQSO_MANAGED işaretini ayarlayın **MQSO_RESUME** Kuyruk A 'daki X aboneliğini sürdürür ve Hobj 'ı Kuyruk A' ya döndürür **MQSO_ALTER** A Kuyruğu A 'da abonelik X 'i sürdürür, Hobj 'ı Kuyruk A' ya döndürür ve izin verilen değişiklikleri yapar	**MQSO_CREATE** RC = 2019-Hatalı nesne tanıtıcısı: yönetilen abonelik yaratmak ve yönetilen bir kuyruk yaratmak için MQSO_MANAGED işaretini ayarlayın **MQSO_RESUME** Dönüş kodu = 2428-X aboneliği yok **MQSO_ALTER** RC = 2019-Hatalı nesne tanıtıcısı: Kuyruk A ya da B yok

Not: Sıkıştırılmış kodlama biçemi, üretim kullanımına hazır olmamaya yöneliktir.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>

void inquireQname(MQHCONN HConn, MQHOBJ Hobj, MQCHAR48 qName);

int main(int argc, char **argv)
{
    MQCHAR48 topicNameDefault            = "STOCKS";
    char     topicStringDefault[]        = "IBM/PRICE";
    char     subscriptionNameDefault[]   = "IBMSTOCKPRICESUB";
    char     subscriptionQueueDefault[]  = "STOCKTICKER";
    char     publicationBuffer[101];      /* Allocate to receive messages     */
    char     resTopicStrBuffer[151];      /* Allocate to resolve topic string */
    MQCHAR48 qmName = "";                 /* Default queue manager            */
    MQCHAR48 qName = "";                  /* Allocate storage for MQINQ       */

    MQHCONN  Hconn = MQHC_UNUSABLE_HCONN; /* connection handle                */
    MQHOBJ   Hobj = MQHO_NONE;            /* subscription queue handle        */
    MQHOBJ   Hsub = MQSO_NONE;            /* subscription handle              */
    MQLONG   CompCode = MQCC_OK;          /* completion code                  */
    MQLONG   Reason = MQRC_NONE;          /* reason code                      */
    MQLONG   messlen = 0;
    MQOD     od = {MQOD_DEFAULT};         /* Unmanaged subscription queue     */
    MQSD     sd = {MQSD_DEFAULT};         /* Subscription Descriptor          */
    MQMD     md = {MQMD_DEFAULT};         /* Message Descriptor               */
    MQGMO   gmo = {MQGMO_DEFAULT};        /* get message options              */
    MQLONG   sdOptions = MQSO_CREATE | MQSO_RESUME | MQSO_DURABLE |
MQSO_FAIL_IF_QUIESCING;

    char *   topicName = topicNameDefault;
    char *   topicString = topicStringDefault;
    char *   subscriptionName =  subscriptionNameDefault;
    char *   subscriptionQueue = subscriptionQueueDefault;
    char *   publication = publicationBuffer;
    char *   resTopicStr = resTopicStrBuffer;
    memset(resTopicStrBuffer, 0, sizeof(resTopicStrBuffer));
```

Şekil 47. Yönetilmeyen MQ abonesi-kısım 1: bildirimler.

```
    switch(argc){                   /* Replace defaults with args if provided */
default:
        switch((argv[5][0])) {
case('A'):  sdOptions =  MQSO_ALTER  | MQSO_DURABLE | MQSO_FAIL_IF_QUIESCING;
            break;
case('C'):  sdOptions =  MQSO_CREATE | MQSO_DURABLE | MQSO_FAIL_IF_QUIESCING;
            break;
case('R'):  sdOptions =  MQSO_RESUME | MQSO_DURABLE | MQSO_FAIL_IF_QUIESCING;
            break;
default:    ;
        }
case(5):
    if (strcmp(argv[4],"/")) /* "/" invalid = No subscription  */
        subscriptionQueue = argv[4];
    else {
        *subscriptionQueue = '\0';
        if (argc > 5) {
            if (argv[5][0] == 'C') {
                sdOptions = sdOptions + MQSO_MANAGED;
            }
        }
        else
            sdOptions = sdOptions + MQSO_MANAGED;
    }

case(4):
    if (strcmp(argv[3],"/")) /* "/" invalid = No subscription  */
        subscriptionName = argv[3];
    else {
        *subscriptionName = '\0';
        sdOptions = sdOptions - MQSO_DURABLE;
    }
case(3):
    if (strcmp(argv[2],"/")) /* "/" invalid = No topic string  */
        topicString = argv[2];
    else
        *topicString = '\0';
case(2):
    if (strcmp(argv[1],"/")) /* "/" invalid = No topic object  */
        topicName = argv[1];
    else
        *topicName = '\0';
case(1):
    sd.Options = sdOptions;
    printf("Optional parameters: "
    printf("topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|C(reate)|
R(esume)\n");
    printf("Values \"%-.48s\"  \"%s\" \"%s\"  \"%-.48s\" sd.Options=%d\n",
           topicName, topicString, subscriptionName, subscriptionQueue, sd.Options);
    }
```

Şekil 48. Yönetilmeyen MQ abonesi-bölüm 2: parametre işleme.

Bu örnekteki parametre işleme ile ilgili ek açıklamalar aşağıdaki gibidir:

switch((argv[5][0]))
Örnekte varsayılan olarak kullanılan MQSUB seçeneği ayarının geçersiz kılınmanın etkisini sınamak için, 5. parametredeki Alter | C reate | Resume ' e giriş seçeneğiniz vardır. Örnek tarafından kullanılan varsayılan ayar MQSO_CREATE | MQSO_RESUME | MQSO_DURABLE' dir.

Not: Setting MQSO_ALTER or MQSO_RESUME without setting MQSO_DURABLE is an error, and sd.SubName must be set and refer to a subscription that can be resumed or altered.

***subscriptionQueue = '\0';**
sdOptions = sdOptions + MQSO_MANAGED;
Varsayılan abonelik kuyruğu, STOCKTICKER değeri MQSO_CREATE olduğu sürece boş değer dizgisiyle değiştirilirse, örnek MQSO_MANAGED işaretini ayarlar ve dinamik bir abonelik kuyruğu oluşturur. Alter or Resume beşinci değiştirgede ayarlandıysa, örneğin davranışı subscriptionNamedeğerine bağlıdır.

***subscriptionName = '\0';**
sdOptions = sdOptions - MQSO_DURABLE;
Varsayılan abonelik IBMSTOCKPRICESUB ise, boş değerli bir dizgiyle değiştirilirse, örnek MQSO_DURABLE işaretini kaldırır. Diğer parametrelere ilişkin varsayılan değerleri sağlayan örneği çalıştırırsanız, STOCKTICKER ' a gönderilen bir ek geçici abonelik oluşturulur ve yinelenen yayınlar alır. Bir sonraki örneği çalıştırıyorsanız, herhangi bir parametre olmadan, yalnızca bir yayın alırsınız.

```
    do {
        MQCONN(qmName, &Hconn, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        if (strlen(subscriptionQueue)) {
            strncpy(od.ObjectName, subscriptionQueue, MQ_Q_NAME_LENGTH);
            MQOPEN(Hconn, &od, MQOO_INPUT_AS_Q_DEF | MQOO_FAIL_IF_QUIESCING | MQOO_INQUIRE,
                  &Hobj, &CompCode, &Reason);
            if (CompCode != MQCC_OK) break;
        }
        strncpy(sd.ObjectName, topicName, MQ_TOPIC_NAME_LENGTH);
        sd.ObjectString.VSPtr = topicString;
        sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
        sd.SubName.VSPtr = subscriptionName;
        sd.SubName.VSLength = MQVS_NULL_TERMINATED;
        sd.ResObjectString.VSPtr = resTopicStr;
        sd.ResObjectString.VSBufSize = sizeof(resTopicStrBuffer)-1;
        MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        gmo.Options = MQGMO_WAIT | MQGMO_NO_SYNCPOINT | MQGMO_CONVERT;
        gmo.WaitInterval = 10000;
        gmo.MatchOptions = MQMO_MATCH_CORREL_ID;
        memcpy(md.CorrelId, sd.SubCorrelId, MQ_CORREL_ID_LENGTH);
        inquireQname(Hconn, Hobj, qName);
        printf("Waiting %d seconds for publications matching \"%s\" from %-0.48s\n",
               gmo.WaitInterval/1000, resTopicStr, qName);
        do {
            memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
            memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
            md.Encoding       = MQENC_NATIVE;
            md.CodedCharSetId = MQCCSI_Q_MGR;
            MQGET(Hconn, Hobj, &md, &gmo, sizeof(publication), publication, &messlen,
&CompCode, &Reason);
            if (Reason == MQRC_NONE)
                printf("Received publication \"%s\"\n", publication);
        }
        while (CompCode == MQCC_OK);
        if (CompCode != MQCC_OK && Reason != MQRC_NO_MSG_AVAILABLE) break;
        MQCLOSE(Hconn, &Hsub, MQCO_NONE, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
        if (CompCode != MQCC_OK) break;
        MQDISC(&Hconn, &CompCode, &Reason);
    } while (0);
    printf("Completion code %d and Return code %d\n", CompCode, Reason);
}
void inquireQname(MQHCONN Hconn, MQHOBJ Hobj, MQCHAR48 qName) {
#define _selectors 1
#define _intAttrs 1

    MQLONG select[_selectors] = {MQCA_Q_NAME}; /* Array of attribute selectors    */
    MQLONG intAttrs[_intAttrs];            /* Array of integer attributes           */
    MQLONG CompCode, Reason;
    MQINQ(Hconn, Hobj, _selectors, select, _intAttrs, intAttrs, MQ_Q_NAME_LENGTH, qName,
&CompCode, &Reason);
    if (CompCode != MQCC_OK) {
        printf("MQINQ failed with Condition code %d and Reason %d\n", CompCode, Reason);
        strncpy(qName, "unknown queue", MQ_Q_NAME_LENGTH);
    }
    return;
}
```

Şekil 49. Yönetilmeyen MQ abonesi-bölüm 3: kod gövdesi.

Bu örnekteki kodla ilgili ek açıklamalar aşağıdaki gibidir:

if (strlen(subscriptionQueue))
If there is no subscription queue name then the example uses MQHO_NONE as the value of Hobj.

MQOPEN(...);
Abonelik kuyruğu açılır ve kuyruk tanıtıcısı Hobjiçinde saklanır.

MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
Abonelik, MQOPEN (ya da abonelik kuyruğu adı yoksa MQHO_NONE) tarafından geçirilen Hobj kullanılarak açılır. An unmanaged queue can be resumed without explicitly opening it with an MQOPEN.

MQCLOSE(Hconn, &Hsub, MQCO_NONE, &CompCode, &Reason);
Abonelik, abonelik tanıtıcısı kullanılarak kapatılır. Aboneliğin dayanıklı olup olmadığına bağlı olarak, abonelik örtük bir MQCO_KEEP_SUB ya da MQCO_REMOVE_SUBile kapatılır. You can close a durable subscription with MQCO_REMOVE_SUB, but you *olamaz* close a non-durable subscription with MQCO_KEEP_SUB. MQCO_REMOVE_SUB işlemi, abonelik kuyruğuna gönderilmekte olan başka yayınların durdurulacağı aboneliği kaldırmaktan başka bir işlem değildir.

MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
Abonelik yönetilmezse, özel bir işlem yapılmamaktadır. Kuyruk yönetilirse ve abonelik belirtik ya da örtük MQCO_REMOVE_SUBile kapatılırsa, tüm yayınlar bu noktada silinerek kuyruktan silinmektedir.

gmo.MatchOptions = MQMO_MATCH_CORREL_ID;
memcpy(md.CorrelId, sd.SubCorrelId, MQ_CORREL_ID_LENGTH);
Alınan iletilerin, aboneliğimiz için geçerli olduğundan emin olun.

Bu örnekten elde edilen sonuçlar, yayınlama/abone olma konularının özelliklerini gösterir:

In Şekil 50 sayfa 287 the example starts by publishing 130 on the NYSE/IBM/PRICE topic.

```
W:\Subscribe3\Debug>..\..\Publish2\Debug\publishstock
Provide parameters:  TopicObject TopicString Publication
Publish "130" to topic "STOCKS" and topic string "IBM/PRICE"
Published "130" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0
```

Şekil 50. 130-NYSE/IBM/PRICE ' yi yayınlayın

Örneğin, örneğe ilişkin Şekil 51 sayfa 287 uygulamasında varsayılan parametreleri kullanarak alıkonan yayın 130olur. Sağlanan konu nesnesi ve konu dizesi, Şekil 55 sayfa 288içinde gösterildiği şekilde yoksayılır. Konu nesnesi ve konu dizgisi, her zaman bir abonelik nesnesinden alınır, biri sağlandığında ve konu dizgisi değişmez. Örneğin, gerçek davranışı MQSO_CREATE, MQSO_RESUMEve MQSO_ALTER ' in seçimine ya da bileşimine bağlıdır. Bu örnekte MQSO_RESUME , seçilen seçenektir.

```
W:\Subscribe3\Debug>solution3
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|
C(reate)|R(esume)
Values "STOCKS"  "IBM/PRICE" "IBMSTOCKPRICESUB"  "STOCKTICKER" sd.Options=8206
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from STOCKTICKER
Received publication "130"
Completion code 0 and Return code 0
```

Şekil 51. Alıkonan yayını al

(Şekil 52 sayfa 287) içinde hiçbir yayın alınmadığından, sürekli abonelik alıkonan yayını zaten almış olabilir. Bu örnekte, abonelik, kuyruk adı olmadan yalnızca abonelik adı sağlanarak sürdürülür. Kuyruk adı sağlandıysa, önce kuyruk açılacaktır ve tanıtıcı MQSUB' a iletilecektir.

Not: The 2038 error from MQINQ is due to the implicit MQOPEN of STOCKTICKER by MQSUB not including the MQOO_INQUIRE option. Avoid the 2038 return code from MQINQ by opening the queue explicitly.

```
W:\Subscribe3\Debug>solution3 STOCKS IBM/PRICE IBMSTOCKPRICESUB / Resume
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|
C(reate)|R(esume)
Values "STOCKS"  "IBM/PRICE" "IBMSTOCKPRICESUB"  "" sd.Options=8204
MQINQ failed with Condition code 2 and Reason 2038
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from unknown queue
Completion code 0 and Return code 0
```

Şekil 52. Aboneliği sürdür

Şekil 53 sayfa 288 içinde örnek, hedef olarak STOCKTICKER kullanarak, kalıcı olmayan bir yönetilmeyen abonelik yaratır. Bu yeni bir abonelik olduğu için alıkonan yayınını alır.

```
W:\Subscribe3\Debug>solution3 STOCKS IBM/PRICE / STOCKTICKER Create
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|
C(reate)|R(esume)
Values "STOCKS"  "IBM/PRICE" ""  "STOCKTICKER" sd.Options=8194
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from STOCKTICKER
Received publication "130"
Completion code 0 and Return code 0
```

Şekil 53. Alıkonan yayını, yönetilmeyen yeni kalıcı olmayan abonelikle al

Şekil 54 sayfa 288 ' ta çakışan abonelikleri göstermek için başka bir yayın gönderilir ve alıkonan yayın değiştiriliyor. Daha sonra, yeni bir kalıcı olmayan, yönetilmeyen bir abonelik, abonelik adı sağlamayarak yaratılır. Alıkonan yayın, yeni abonelik için iki kez, yeni abonelik için bir kez ve STOCKTICKER kuyruğunda hala etkin olan kalıcı IBMSTOCKPRICESUB aboneliği için bir kez alınır. Örnek, uygulamanın değil, abonelikleri olan kuyruğun olduğu bir resimdir. Uygulamanın bu çağrısındaki IBMSTOCKPRICESUB aboneliğine başvurmamasına rağmen, uygulama yayını iki kez alır: bir kez yönetimsel olarak oluşturulan sürekli abonelikten, bir kez de uygulamanın kendisi tarafından oluşturulan kalıcı olmayan abonelikten.

```
W:\Subscribe3\Debug>..\..\Publish2\Debug\publishstock
Provide parameters:  TopicObject TopicString Publication
Publish "130" to topic "STOCKS" and topic string "IBM/PRICE"
Published "130" to topic string "NYSE/IBM/PRICE"
Completion code 0 and Return code 0

W:\Subscribe3\Debug>solution3 STOCKS IBM/PRICE / STOCKTICKER Create
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|
C(reate)|R(esume)
Values "STOCKS"  "IBM/PRICE" ""  "STOCKTICKER" sd.Options=8194
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from STOCKTICKER
Received publication "130"
Received publication "130"
Completion code 0 and Return code 0
```

Şekil 54. Çakışma abonelikleri

Şekil 55 sayfa 288 ' ta örnek, yeni bir konu dizgisi sağlamanın ve var olan bir aboneliğin değiştirilme aboneliğiyle sonuçlanmadığını gösterir.

1. İlk durumda, `Resume` , beklediğiniz gibi, var olan aboneliğe devam eder ve değiştirilen konu dizesini yoksayar.
2. İkinci durumda `Alter` , bir hataya neden olur, `RC = 2510, Topic not alterable`.
3. Üçüncü örnekte `Create` , bir hataya neden olur `RC = 2432, Sub already exists`.

```
W:\Subscribe3\Debug>solution3 "" NASDAC/IBM/PRICE IBMSTOCKPRICESUB STOCKTICKER Resume
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|C(reate)|R(esume)
Values ""  "NASDAC/IBM/PRICE" "IBMSTOCKPRICESUB"  "STOCKTICKER" sd.Options=8204
Waiting 10 seconds for publications matching "NYSE/IBM/PRICE" from STOCKTICKER
Received publication "130"
Completion code 0 and Return code 0

W:\Subscribe3\Debug>solution3 "" NASDAC/IBM/PRICE IBMSTOCKPRICESUB STOCKTICKER Alter
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|C(reate)|R(esume)
Values ""  "NASDAC/IBM/PRICE" "IBMSTOCKPRICESUB"  "STOCKTICKER" sd.Options=8201
Completion code 2 and Return code 2510

W:\Subscribe3\Debug>solution3 "" NASDAC/IBM/PRICE IBMSTOCKPRICESUB STOCKTICKER Create
Optional parameters: topicName, topicString, subscriptionName, subscriptionQueue, A(lter)|C(reate)|R(esume)
Values ""  "NASDAC/IBM/PRICE" "IBMSTOCKPRICESUB"  "STOCKTICKER" sd.Options=8202
Completion code 2 and Return code 2432
```

Şekil 55. Abonelik konuları değiştirilemez

İlgili kavramlar

“Örnek 1: MQ Publication consumer” sayfa 274
MQ Publication tüketicisi, konuların kendisine abone olmayan bir IBM WebSphere MQ ileti tükettidir.

“Örnek 2: Yönetilen MQ abonesi” sayfa 276

Yönetilen MQ abonesi, çoğu abone uygulaması için tercih edilen kalıbdır. Örneğin, kuyrukların *hayır* yönetim tanımlaması, konuları ya da abonelikleri gerektirir.

"Yayınlayıcı uygulamaları yazılıyor" sayfa 267
İki örnek üzerinde çalışarak yayınlayıcı uygulamaları yazmaya başlayın. İlki, bir kuyruğa ileti yerleştirmek için bir noktadan noktaya mümkün olduğunca yakın bir şekilde modellenir ve ikinci olarak konular, yayınlayıcı uygulamaları için dinamik olarak daha yaygın bir kalıba yol açmaktadır.

Yaşam çevrimlerini yayınla/abone ol

Yayınlama/abone olma uygulamaları tasarlarken konuların, aboneliklerin, abonelerin, yayınların, yayıncıların ve kuyrukların yaşam çevrimlerini göz önünde bulundurun.

Bir nesnenin (abonelik gibi) yaşam çevrimi, yaratılımıyla başlar ve silme işlemi ile sona erer. Ayrıca geçici askıya alma, üst ve alt konular olması, süre bitimi ve silinme gibi diğer durumları ve geçeceği değişiklikleri de içerebilir.

Geleneksel olarak, kuyruklar gibi geleneksel WebSphere MQ nesneleri yönetimsel olarak ya da Programlanır Komut Biçimi (PCF) kullanan yönetimsel programlar tarafından yaratılır. Yayınlama/abone olma, MQSUB ve MQCLOSE API fiillerinin, yalnızca kuyrukları yaratmayan ve silmeyen, ancak tüketilmeyen iletileri temizleyen ve yönetimsel olarak oluşturulan konu nesneleri ile programsal olarak ya da yönetimsel olarak konu dizgileri arasında ilişkilendirmeler içeren yönetilen abonelikler kavramına sahip olan abonelikleri oluşturmak ve silmek için farklı bir şekilde yayınlanabilir/abone olunması farklıdır.

Bu işlevsel zenginlik, geniş bir yayın/abone olma gereksinimleri yelpazesi için hizmet sağlar ve aynı zamanda yayınlama/abone olma uygulamasının bazı ortak kalıplarını tasarlamayı basitleştirir. Yönetilen abonelikler, örneğin, yalnızca bu aboneliği yaratan program kadar uzun bir süre için amaçlanan bir aboneliğin hem programlamasını hem de yönetimini basitleştirir. Yönetilmeyen abonelikler, abone olmak ve yayınları tüketmek arasında gevşek bir bağlantının olduğu programlamayı basitleştirir. Merkezi olarak oluşturulan abonelikler, örüntünün merkezi bir denetim modeli (örneğin, uçuş bilgilerini otomatik geçitlere göndermesi gibi) merkezi bir denetim modeline dayalı olarak, tüketicilere yönlendirici yayın trafiğinden biri olduğu durumlarda kullanışlıdır, ancak geçit personeli bir kapıya bir uçuş numarası girerek, uçuş için yolcular kayıtlarına abone olmak üzere programlı olarak oluşturulmuş abonelikler kullanılırmış.

bu son örnekte yönetilen bir dayanıklı abonelik uygun olabilir: yönetilen, abonelikler çok sık oluşturulmaya başlandığından ve geçit kapandığında ve abonelik programlı olarak kaldırıldığında net bir uç noktası olması, bir nedenle veya başka bir nedenle aşağı giden geçit abone programı nedeniyle bir yolcu kaydını kaybetmemek için dayanıklı bir uç nokta olmalıdır.[2]. yolcu kayıtlarının geçitten yayınlanmasını başlatmak için, olası bir tasarım, geçit numarası kullanılarak hem yolcu kayıtlarına abone olmak için geçit uygulaması için hem de geçit numarasını kullanarak geçit açma olayını yayınlayacaktır. yayıncı, yolcu kayıtlarını yayınlayarak geçit açma olayına cevap veriyor. daha sonra faturalama gibi diğer ilgili taraflara da gidilebilecek olan uçuş kayıtlarını, uçuşun yerini ve müşteri hizmetlerine, kapı numarasının yolcuların cep telefonlarına metin bildirimlerine (mesaj) bildirerek yanıt veriyor.

Merkezi olarak yönetilen abonelik, her bir geçit için önceden tanımlanmış bir kuyruk kullanarak, dayanıklı, yönetilmeyen bir model, yönlendirme yolcu listelerini kapıya yönlendirebilir.

Sonraki yayınlama/abone olma yaşam çevrimlerine ilişkin aşağıdaki üç örnek, kalıcı olmayan, yönetilen dayanıklı ve yönetilmeyen dayanıklı abonelerin abonelikler, konular, kuyruklar, yayıncılar ve kuyruk yöneticisi ile nasıl etkileşimde bulunabileceğini ve sorumlulukların yönetim ile abone programları arasında nasıl bölünebileceğini gösterir.

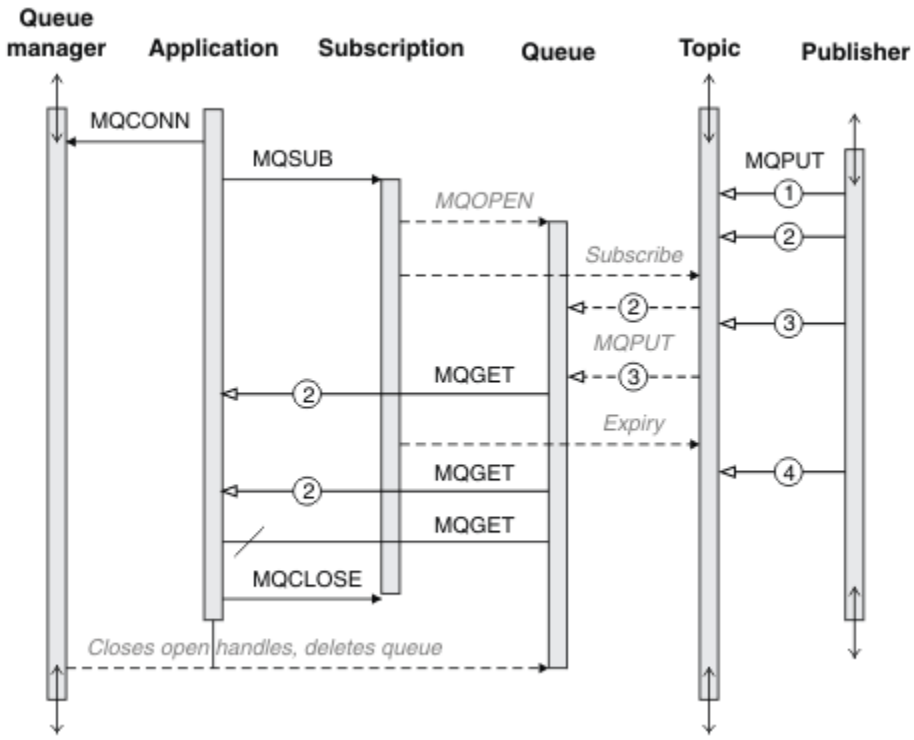
Yönetilen kalıcı olmayan abone

Şekil 56 sayfa 290 , yönetilen bir kalıcı olmayan abonelik yaratan bir uygulamayı gösterir; abonelikte belirtilen konuya ilişkin iki ileti yayınlanır ve sonlandırılır. Noktalı oklu italik gri yazı tipiyle etiketlenen etkileşimler örtük olur.

Notların bir kısmı var.

[2] Yayıncı, diğer olası arızaları önlemek için yolcu kayıtlarını kalıcı iletiler olarak göndermelidir, elbette.

1. Uygulama, zaten iki kez yayınlanmış bir konuda abonelik yaratır. Abone ilk yayınını aldığında, alıkonan yayın olan *saniye* yayınını alır.
2. Kuyruk yöneticisi, geçici bir abonelik kuyruğu yaratmanın yanı sıra, konu için bir abonelik yaratmanın yanı sıra, bir geçici abonelik kuyruğu da yaratır
3. Aboneliğin süre bitimi var. Aboneliğin süresi dolduğunda, bu abonelikle ilgili yayınların gönderilmesi sona ermez, ancak abone aboneliğin sona ermesinden önce yayınlanan iletileri almaya devam eder. Yayın süre bitimi, abonelik süre bitiminden etkilenmez.
4. Dördüncü yayın abonelik kuyruğuna konmaz ve sonuç olarak son MQGET yayını geri döndürmez.
5. Abone aboneliğini kapattısa da, kuyrukla ya da kuyruk yöneticisiyle olan bağlantısını kapatmıyor.
6. Uygulama sona erdikten kısa bir süre sonra kuyruk yöneticisi temizliyor. Abonelik yönetiliyor ve kalıcı olmayan bir abonelik kuyruğu olduğundan, abonelik kuyruğu silinir.



Şekil 56. Yönetilen kalıcı olmayan abone ömür çizgileri

Yönetilen dayanıklı abone

Yönetilen dayanıklı abone, önceki örneği bir adım daha ileriye götürür ve abone olunan uygulamanın sona erdirilmesinin ve yeniden başlatılıp başlatılabilmesinin bir yönetilen aboneliğini gösterir.

Notlara dikkat etmek için yeni noktalar var.

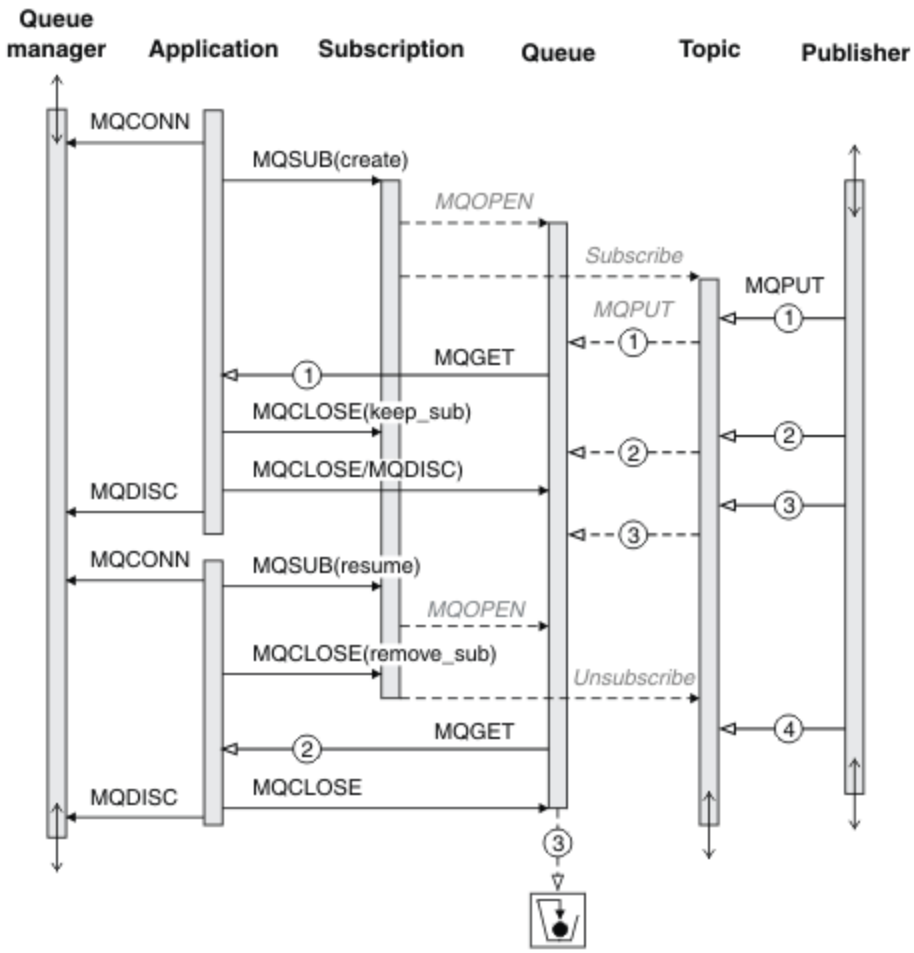
1. Bu örnekte, son olarak, yayın konusu abonelikte tanımlanmadan önce var olmamıştı.
2. The first time the subscriber terminates, it closes the subscription with the option MQCO_KEEP_SUB. Yönetilen bir kalıcı aboneliği örtük olarak kapamak için varsayılan davranış budur.
3. Abone aboneliğe devam ettiğinde, abonelik kuyruğu yeniden açılır.
4. Yeniden açılmadan önce kuyruğa yerleştirilen yeni yayın 2, abonelik kaldırıldıktan sonra da MQGET' e kullanılabilir.

   Abonelik dayanıklı olsa da, abonenin gönderdiği tüm iletiler, yalnızca *her ikisi* abonelik dayanıklı ve iletiler kalıcı olduğunda aboneye güvenir. Message persistence depends on the setting of the

Persistent field in the MQMD of the message sent by the publisher. Bir abonenin bu konuda bir kontrolü yok.

5. MQCO_REMOVE_SUB işaretiyle aboneliği kapatmak, aboneliği kaldırır ve abonelik kuyruğuna yerleştirilen diğer yayınların durdurulmasına neden olur. When the subscription queue is closed, then the queue manager removes the unread publication 3, and then deletes the queue. İşlem, aboneliğin yönetimsel olarak silinmesine eşdeğerdir.

   Not: Do not delete the queue manually, or issue MQCLOSE with the option MQCO_DELETE, or MQCO_PURGE_DELETE. Yönetilen bir aboneliğin görünür somutlama ayrıntıları, desteklenen WebSphere MQ arabiriminin bir parçası değildir. Kuyruk yöneticisi yönetimi, tam denetimi olmadığı sürece aboneliği güvenilir bir şekilde yönetemez.



Şekil 57. Yönetilen dayanıklı abone ömür çizgileri

Yönetilmeyen dayanıklı abone

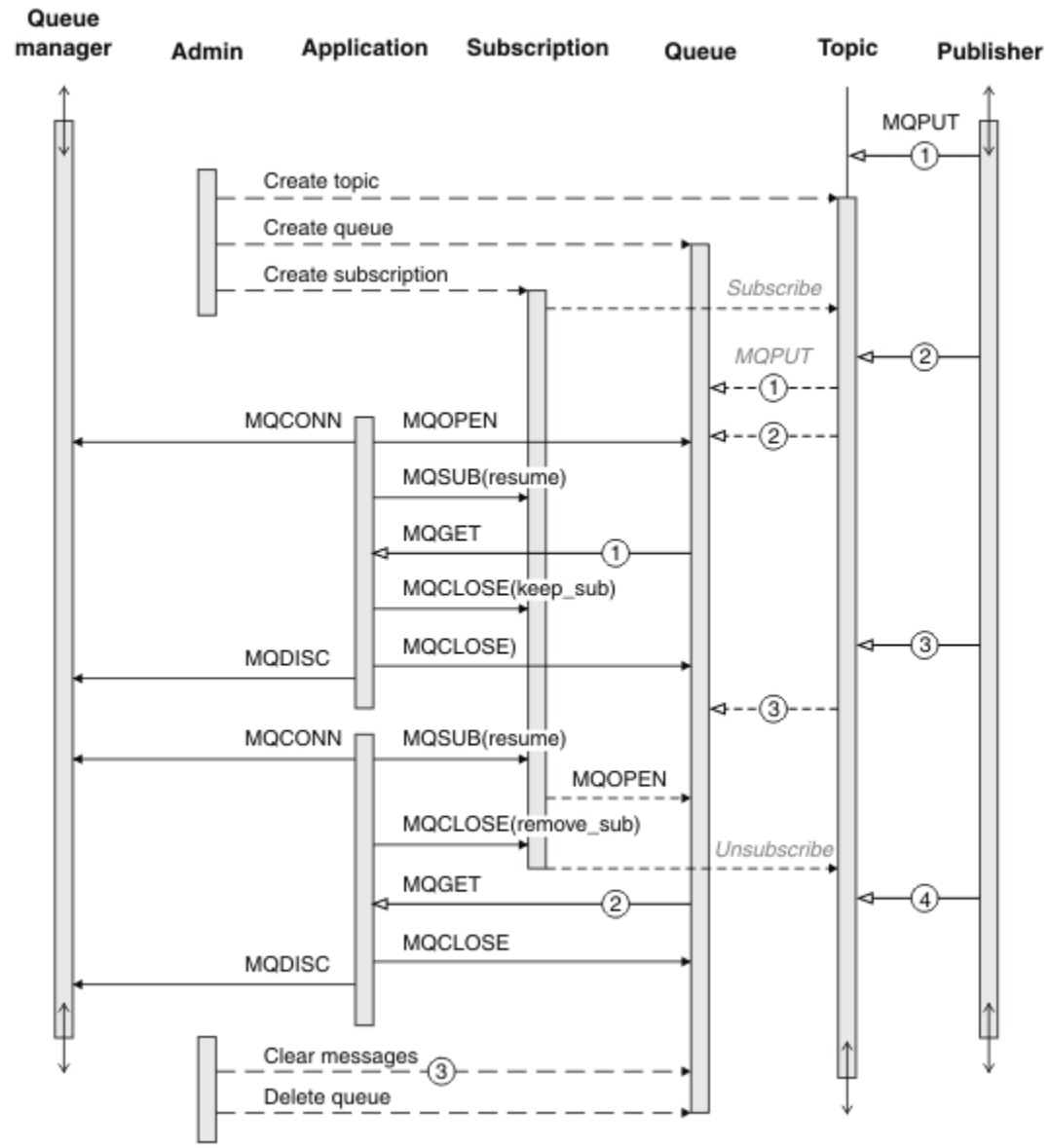
Üçüncü örneğe bir yönetici eklenir: yönetilmeyen dayanıklı abone. Yöneticinin bir yayınlama/abone olma uygulaması ile nasıl etkileşimde bulunabileceğini göstermek için iyi bir örnektir.

Notların olduğu noktalar listelenir.

1. Yayınlayıcı bir iletiyi (1), daha sonra abonelik için kullanılan konu nesnesiyle ilişkilendirilecek bir konuya koyar. Konu nesnesi, genel arama karakterleri kullanılarak yayınlanan konuyla eşleşen bir konu dizgisini tanımlar.
2. Konu, korunan bir yayınına sahip.

3. Yönetici bir konu nesnesi, bir kuyruk ve bir abonelik yaratır. Konu nesnesi ve kuyruğun abonelikten önce tanımlanması gerekir.
4. Uygulama, abonelikle ilişkili kuyruğu açar ve MQSUB ' un kuyruğun tanıtıcısını geçirmesini sağlar. It could, alternatively, simply open the subscription, passing it the queue handle MQHO_NONE. Converse doğru değil, yalnızca kuyruk tanıtıcısını abonelik adı olmadan geçirerek bir aboneliği sürdüremez; bir kuyruk birden çok aboneliğine sahip olabilir.
5. Uygulama, aboneliği ilk kez açmış olsa da, MQSO_RESUME seçeneğini kullanarak aboneliği açar. Bir yönetimsel olarak oluşturulan bir aboneliğin sürdürülmesi.
6. Abone alıkonan yayını alır, 1. Publication 2, although published before any publications were received by the subscriber, was published after the subscription started, and is the second publication on the subscription queue.

   Not: Alıkonan yayın kalıcı bir ileti olarak yayınlanmadıysa, kuyruk yöneticisi yeniden başlatıldıktan sonra bu yayın kaybedilir.
7. Bu örnekte, abonelik dayanıklıdır. Bir programın yönetilmeyen, kalıcı olmayan bir abonelik yaratması mümkündür; bu, yöneticinin yapabildiği bir şey olmadığı açık olmalıdır.
8. The effect of the option MQCO_REMOVE_SUB on closing the subscription is to remove the subscription just as if the administrator had deleted it. Bu, kuyruğa gönderilen tüm yayınları durdurur; ancak, kuyruk kapatıldığında bile, önceden kuyrukta olan yayınları etkilemez; *yönetilen* kalıcı bir aboneliğin tersine.
9. Yönetici daha sonra geri kalan iletiyi (3) siler ve kuyruğu siler.



Şekil 58. Yönetilmeyen dayanıklı abone ömür çizgileri

Yönetilmeyen bir abonelik için normal bir kalıp, yönetici tarafından gerçekleştirilecek kuyruk ve abonelik ev bakımı içindir. Tipik olarak, bir yönetilen abonenin davranışını taklit etmeyi ve uygulama kodunda programsal olarak kuyruklar ve abonelikler toplamayı denemez. Yönetim mantığını yazmaya gerek duyuyorsanız, yönetilen bir kalıp kullanarak aynı sonuçları elde edip edemediğinizi sorgunuz. Sıkı bir şekilde senkronize, tamamen güvenilir bir yönetim kodu yazmak kolay değildir. İletilerin, aboneliklerin ve kuyrukların durumundan bağımsız olarak silinebilmesi için, daha sonra el ile ya da otomatik bir yönetim programı kullanarak daha sonra toparlanabilirsiniz.

İleti özelliklerini yayınla/abone ol

Birçok ileti özelliği, WebSphere MQ yayınlama/abone olma mesajlarıyla ilgilidir.

PubAccountingSimgesi

Bu, bu abonelikle eşleşen tüm yayın iletilerinin İleti Tanımlayıcısının (MQMD) AccountingToken alanında yer alacak değerdir. AccountingToken , iletinin kimlik bağlamının bir parçasıdır. İleti bağlamına ilişkin daha

fazla bilgi için bkz. “İleti bağlamı” sayfa 37. MQMD ' deki AccountingToken alanı hakkında daha fazla bilgi için bkz. AccountingToken.

PubApplIdentityData

Bu, bu abonelikle eşleşen tüm yayın iletilerinin (MQMD) ApplIdentityVeri alanında yer alacak değerdir. ApplIdentityVerileri, iletinin kimlik bağlamının bir parçasıdır. İleti bağlamına ilişkin daha fazla bilgi için bkz. “İleti bağlamı” sayfa 37. MQMD ' de ApplIdentityVeri alanıyla ilgili daha fazla bilgi için bkz. ApplIdentityData.

MQSO_SET_IDENTITY_CONTEXT seçeneği belirtilmediyse, bu abonelik için yayınlanan her iletide ayarlanacak olan ApplIdentityVerisi, varsayılan bağlam bilgileri olarak boşluklara sahip olur.

MQSO_SET_IDENTITY_CONTEXT seçeneği belirtilirse, kullanıcı tarafından PubApplIdentityData oluşturulmakta ve bu alan, bu abonelik için her yayında ayarlanacak ApplIdentityVerilerini içeren bir giriş alanıdır.

PubPriority

Bu değer, bu abonelikle eşleşen tüm yayın iletilerinin İleti Tanımlayıcısının (MQMD) Öncelik alanında olacak değer. MQMD ' deki Öncelik alanıyla ilgili daha fazla bilgi için Priority(Öncelik) başlıklı konuya bakın.

Değer sıfırdan büyük ya da sıfıra eşit olmalıdır; sıfır, en düşük önceliğe sahip olmalıdır. Aşağıdaki özel değerler de kullanılabilir:

- MQPRI_PRIORITY_AS_Q_DEF-Bir abonelik kuyruğu, MQSUB çağrısındaki Hobj alanında sağlandığında ve yönetilen bir tanıtıcı değilse, iletinin önceliği bu kuyruğun DefProirity özniteliğinden alınır. Belirlenen kuyruk bir küme kuyruğunsa ya da kuyruk-adı çözünürlük yolunda birden çok tanımlama varsa, bu öncelik MQMD ' deki Öncelik için açıklandığı gibi, yayın iletisi kuyruğa konduğunda öncelik belirlenir. MQSUB çağrısı yönetilen bir tanıtıcı kullanıyorsa, ileti için öncelik, abone olunan konumla ilişkili model kuyruğunun DefPriority özniteliğinden alınır.
- MQPRI_PRIORITY_AS_PUBLICID-İletiye ilişkin öncelik, özgün yayının önceliğidir. Bu, bu alanın ilk değeridir.

SubCorrelTanıtıcısı

Uyarı: Bir ilinti tanıtıcısı yalnızca, bir sıradüzeninde değil, yayınlama/abone olma kümesindeki kuyruk yöneticileri arasında geçirilebilir.

Bu abonelikle eşleşmesi için gönderilen tüm yayınlar, ileti tanımlayıcısında bu ilinti tanıtıcısını içerir. Birden çok abonelik, yayınlarını almak için aynı kuyruğu kullanıyorsa, ilinti tanıtıcısı temelinde MQGET kullanılması, yalnızca belirli bir abonelik için yayınların elde edilebilmesini sağlar. Bu ilinti tanıtıcısı kuyruk yöneticisi ya da kullanıcı tarafından yaratılabilir.

MQSO_SET_COREL_ID seçeneği belirtilmediyse, ilinti tanıtıcısı kuyruk yöneticisi tarafından oluşturulur ve bu alan, bu abonelik için yayınlanan her iletide ayarlanacak ilinti tanıtıcısını içeren bir çıkış alanıdır.

MQSO_SET_COREL_ID seçeneği belirtilirse, kullanıcı tarafından ilinti tanıtıcısı oluşturulmakta ve bu alan, bu abonelik için her yayında ayarlanacak ilinti tanıtıcısını içeren bir giriş alanıdır. Bu durumda, alan MQCI_NONE içeriyorsa, bu abonelik için yayınlanan her iletide ayarlanacak ilinti tanımlayıcısı, iletinin özgün tanıtıcısıyla yaratılan ilinti tanımlayıcısıdır.

MQSO_GROUP_SUB seçeneği belirtilirse ve belirtilen ilinti tanıtıcısı, aynı kuyruğu ve çakışan bir konu dizesini kullanan var olan gruplanmış bir abonelikle aynıysa, yayının bir kopyasıyla yalnızca gruptaki en önemli abonelik sağlanır.

SubUserVerileri

Bu, abonelik kullanıcı verileridir. Bu alandaki aboneliğe ilişkin sağlanan veriler, bu aboneliğe gönderilen her yayının MQSubUserVeri iletisi özelliği olarak içerilir.

Yayın özellikleri

Çizelge 44 sayfa 295 içinde, bir yayın iletisiyle birlikte sağlanan yayın özellikleri listelenir.

You can access these properties directly from the **MQRFH2** folder, or retrieve them using MQINQMP. MQINQMP , sorgulanacak özelliğin adı olarak özellik adını ya da **MQRFH2** adını kabul eder.

Çizelge 44. Yayın özellikleri

Özellik adı	MQRFH2 adı	Tip	Tanım
MQTopicString	mqps.Top	MQTYPE_STRING	Konu dizesi
MQSubUserVerileri	mqps.Sud	MQTYPE_STRING	Abone kullanıcı verileri
MQIsRetained	mqps.Ret	MQTYPE_BOOLEAN	Alıkonan yayın
MQPubOptions	mqps.Pub	MQTYPE_INT32	Yayın seçenekleri
MQPubLevel	mqps.Pbl	MQTYPE_INT32	Yayın düzeyi
MQPubTime	mqpse.Pts	MQTYPE_STRING	Yayın zamanı
MQPubSeqNum	mqpse.Seq	MQTYPE_INT32	Yayın sıra numarası
MQPubStrIntData	mqpse.Sid	MQTYPE_STRING	Yayıncı tarafından eklenen String/Integer verileri
MQPubFormat	mqpse.Pfmt	MQTYPE_INT32	İleti biçimi: MQRFH1 MQRFH2 PCF

İleti sıralaması

Belirli bir konu için, iletiler kuyruk yöneticisi tarafından, yayınlama uygulamalarından alındıkları sırayla yayınlanır (ileti önceliğine dayalı olarak yeniden sıralamaya tabi olarak).

Olağan durumda ileti sıralaması, her abonenin belirli bir kuyruk yöneticisinden belirli bir konuya, belirli bir yayıncıdan yayınlandığı sırayla yayınlandığı sırayla ileti aldığı anlamına gelir.

Ancak, tüm WebSphere MQ iletileriyle olduğu gibi, iletiler, zaman zaman, siparişin dışında teslim edilebilmek için mümkün olur. Bu durum aşağıdaki durumlarda gerçekleşebilir:

- Ağdaki bir bağlantı aşağı inerse ve sonraki iletiler başka bir bağlantı boyunca yeniden yönlendirilirse
- Bir kuyruk geçici olarak tam olarak dolduysa ya da geçici olarak engellenirse, bir ileti çıkmaz bir kuyruğa konursa ve bu nedenle gecikirse, sonraki iletiler düz olarak geçer.
- Sistem yöneticisi, yayıncılar ve aboneler çalışmaya devam ederken bir kuyruk yöneticisini silerse, kuyruğa alınan iletilerin, durdurulacak ileti kuyruğuna ve aboneliklere konmasına neden olur.

Bu şartlar ortaya çıkmazsa, yayınlar her zaman sırayla teslim edilir.

Not: Publish/Subscribe ile gruplanmış ya da bölümlenmiş iletiler kullanmak olanaklı değildir.

Yayınların ele geçirmesi

Bir yayını durdurabilir, değiştirebilir ve daha sonra başka bir aboneye ulaşmadan yeniden yayınlayabilirsiniz.

Aşağıdaki eylemlerden birini yapmak için bir yayının bir aboneye ulaşmadan önce kesişmesini isteyebilirsiniz:

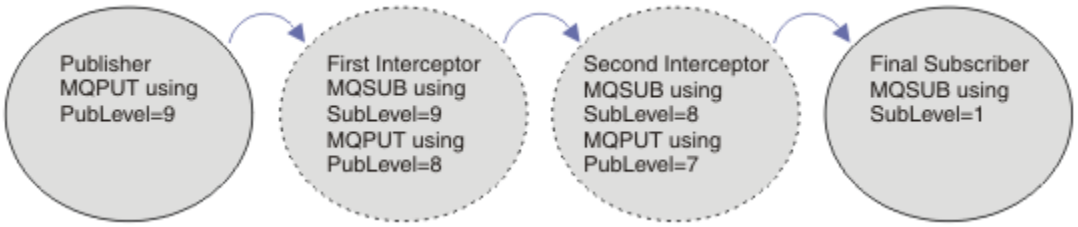
- İletiye ek bilgi ekle

- İletiyi engelle
- İletiyi dönüştürür

Her iletide aynı işlemi gerçekleştirebilir ya da işlemi, aboneliğe, iletiye ya da ileti üstbilgisine bağlı olarak değişiklik yapabilirsiniz.

İlgili başvurular
MQ_PUBLISH_EXIT-Yayınlama çıkışı

Abonelik düzeyleri
Bir yayının son abonelerine ulaşmadan önce kesişmesini engellemeye ilişkin abonelik düzeyini ayarlayın. Abone olan bir abone daha yüksek abonelik düzeyine abone olur ve daha düşük bir yayın düzeyinde yeniden yayınlar. Son abonelere teslim edilmeden önce, bir yayında ileti işleme işlemini gerçekleştirmek için bir araya gelen aboneler zinciri oluşturun.



Şekil 59. Engellenen abonelerin sırası

Bir yayını durdurmak için **MQSD** SubLevel özniteliğini kullanın. Bir ileti algılandıktan sonra, **MQPMO** PubLevel özniteliğini değiştirerek, bu ileti dönüştürülebilir ve daha sonra daha düşük bir yayın düzeyinde yeniden yayınlanabilir. Daha sonra, ileti son abonelere gider ya da daha düşük bir abonelik düzeyinde ara abone tarafından yeniden durdurulanır.

Kesişme abonesi genellikle, yeniden yayınlamadan önce bir iletiyi dönüştürür. Bir dizi kesişme abonesi bir ileti akışı oluşturur. Diğer bir seçenek olarak, kesişen yayını yeniden yayınlamayabilirsiniz: Alt abonelik düzeylerindeki aboneler iletiyi almazlardı.

Diğer abonelerden önce, yayıncının yayınları aldığından emin olun. Engelleyici abonelik düzeyini diğer abonelere göre ayarlayın. Varsayılan olarak, abonelerin bir SubLevel ' si (1) vardır. En yüksek değer 9' dir. Bir yayının en az en yüksek SubLevelen az bir PubLevel (Ortak Düzey)withile başlaması gerekir. Başlangıçta 9' un varsayılan PubLevel ile yayınlayın.

- Bir konuyla ilgili bir aboneye sahipseniz, SubLevel ' i 9olarak ayarlayın.
- Bir konuyla ilgili birden çok kesme uygulaması için, her ardışık işlem abonesi için alt SubLevel ayarlayın.
- You can implement a maximum of 8 intercepting applications, with subscription levels from 9 down to 2 inclusive. İletinin son alıcısının bir SubLevel (1) vardır.

Yayının PubLevel ' a eşit ya da daha düşük en yüksek abonelik düzeyine sahip kesici, önce yayınını alır. Belirli bir abonelik düzeyinde bir konu için tek bir araya girme abonesi yapılandırın. Belirli bir abonelik düzeyinde birden çok abonenin olması, yayının birden çok kopyasında abone olunan uygulamaların son kümesine gönderilmektedir.

A subscriber with a SubLevel of 0 is used as a catchall. Bu ileti, son abonenin iletiyi almaması durumunda yayınını alır. A subscriber with SubLevel of 0 might be used to monitor the publications that no other subscribers received.

Bir araya gelen aboneyi programlamak

Çizelge 45 sayfa 297içinde açıklanan abonelik seçeneklerini kullanın.

Çizelge 45. Abonelerin kesişmesine ilişkin abonelik seçenekleri	
Abonelik seçeneği	**Notlar**
`MQSO_SET_CORREL_ID` ve `SubCorrelId` , `MQCI_NONE`olarak ayarlanır	Kesişen yayının `CorrelId` (CorrelId) yayınını özgün yayınla aynı tutun. **Not:** Bir yayının bir sıradüzeninde ilinti tanıtıcısını geçemezsiniz. Alan, kuyruk yöneticisi tarafından kullanılır.
`PubPriority` , `MQPRI_PRIORITY_AS_PUBLISHED`olarak ayarlanır	Engellenen yayının önceliğini, özgün yayınla aynı tutun.

The options in Çizelge 45 sayfa 297 must be used by all the intercepting subscribers. Sonuçta, ilinti tanıtıcısı ve ileti önceliği, özgün yayınlayıcı ayarından değiştirilmez.

Kesişme abonesi yayını işlediğinde, iletiyi, kendi aboneliğinin `SubLevel` değerinden daha düşük bir `PubLevel` ile aynı konuya yeniden yayınlar. If the intercepting subscriber set a `SubLevel` of 9, it republishes the message with a `PubLevel` of 8.

İletiyi doğru bir şekilde yeniden yayınlamak için, özgün yayından birkaç bilgi parçası gereklidir. Reuse the same **MQMD** as in the original message and set `MQPMO_PASS_ALL_CONTEXT` to ensure all information in the **MQMD** is passed on to the next subscriber. Copy the values from the message properties shown in Çizelge 46 sayfa 297 into the corresponding fields of the republished message. Kesişme abonesi bu değerleri değiştirebilir. İleti koyma seçeneklerini birleştirmek için, **MQPMO**.`Seçenekler` alanına ek değerler eklemek için OR işlecini kullanın.

Yönetilen yayın kuyruğunu kullanmak yerine, yayın kuyruğunu açık bir şekilde açmanız gerekir. You cannot set `MQSO_SET_CORREL_ID` for a managed queue. You also cannot set `MQOO_SAVE_ALL_CONTEXT` on a managed queue. “Örnekler” sayfa 298' ta listelenen kod parçalarına bakın.

Çizelge 46. Yeniden yayınlanan iletiler içinMQPUT değerleri	
İletiyi MQPUT kullanarak yeniden yayınla	**Yayın iletisinde bilgi**
MQOD.`ObjectString`	İleti Özelliği `MQTopicString`
MQPMO.`Options`	İleti Özelliği `MQPubOptions`

Son abonede, abonelik seçeneklerini farklı bir şekilde ayarlama seçeneği vardır. Örneğin, yayınlama önceliğini belirtik olarak `MQPRI_PRIORITY_AS_PUBLISHED`yerine de ayarlayabilirsiniz. Son abonenin ayarları, yalnızca zincirdeki son kesişen aboneden yayını etkiler.

Alıkonan yayınlar

Alıkonan bir yayın, özgün put-message seçeneklerini yeniden yayınlanan iletiye kopyalayarak, araya girdikten sonra korunmalıdır.

`MQPMO_TUT` seçeneği, yayınlayıcı tarafından ayarlanır. Her bir araya gelen abonenin, `MQPubOptions` ' i, yeniden yayınlanan iletinin put-message (put-message) seçeneklerine Çizelge 46 sayfa 297içinde gösterildiği şekilde aktarması gerekir. Put-message seçeneklerinin kopyalanması, özgün yayınlayıcıya göre, yayının korunulup tutulmayacağı gibi seçenekleri korur.

Bir yayın, kesişme abone olan abonelerin geçişini bitirdiğinde ve son abonelere teslim edildiğinde, son abonelere teslim edilir. Yeni aboneler, `SubLevel` 1' da, alıkonan yayını istiyor, daha fazla bir araya getirmeden teslim alıyor. 1 değerinden büyük bir `SubLevel` aboneler, alıkonan yayın gönderilmez. Sonuç olarak, alıkonan yayın, abonelerin ikinci kez kesişme zincirine göre değiştirilmez.

Örnekler

Örnekler, bir araya gelen aboneyi oluşturmak için birleştirilebilen kod parçalarıdır. Kod, üretim kalitesinden çok kısa bir süre olacak şekilde yazılır.

Şekil 60 sayfa 298 içindeki önişlemci yönergelerinde, MQINQMP MQI çağrısı için gerekli olan yayın iletilerinden alınacak iki özellik tanımlanır.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cmqc.h>
#define      MQPUBOPTIONS        (MQPTR)(char*) "MQPubOptions",\
                                 0,\
                                 12,\
                                 MQVS_NULL_TERMINATED,\
                                 MQCCSI_APPL
#define      MQTOPICSTRING       (MQPTR)(char*) "MQTopicString",\
                                 0,\
                                 13,\
                                 MQVS_NULL_TERMINATED,\
                                 MQCCSI_APPL
```

Şekil 60. Ön işlemci yönergeleri

Şekil 61 sayfa 298 , kod parçalarında kullanılan bildirimleri listeler. Vurgulanan terimler dışında, bildirimler bir WebSphere MQ uygulaması için standartdır.

Vurgulanan put ve Al seçenekleri, tüm bağlamı geçirmek için başlatılır. Vurgulanan MQTOPICSTRING ve MQPUBOPTIONS , ön işlemci yönergelerinde tanımlı olan özellik adlarına ilişkin MQCHARV initializuer'larıdır. Adlar MQINQMP' a iletilir.

```
int main(int argc, char **argv) {
    MQLONG   Reason            = MQRC_NONE;
    MQLONG   CompCode          = MQCC_OK;
    MQHCONN  Hcon              = MQHC_UNUSABLE_HCONN;
    MQCHAR   QMName[49]        = " ";
    MQCMHO   CrtMsgHOpts       = {MQCMHO_DEFAULT};
    MQHMSG   Hmsg              = MQHM_NONE;
    MQMD     md                = {MQMD_DEFAULT};
    MQHOBJ   gHobj             = MQHO_NONE;
    MQOD     getOD             = {MQOD_DEFAULT};
    MQGMO    gmo               = {MQGMO_DEFAULT};
    MQLONG   GO_Options        = MQOO_INPUT_AS_Q_DEF
                               | MQOO_FAIL_IF_QUIESCING
                               | MQOO_SAVE_ALL_CONTEXT;
    MQLONG   GC_Options        = MQCO_DELETE_PURGE;
    MQHOBJ   Hsub              = MQHO_NONE;
    MQSD     sd                = {MQSD_DEFAULT};
    MQLONG   SC_Options        = MQCO_NONE;
    MQHOBJ   pHobj             = MQHO_NONE;
    MQOD     putOD             = {MQOD_DEFAULT};
    MQLONG   PO_Options        = MQOO_OUTPUT
                               | MQOO_FAIL_IF_QUIESCING
                               | MQOO_PASS_ALL_CONTEXT;
    MQLONG   PC_Options        = MQCO_NONE;
    MQPMO    pmo               = {MQPMO_DEFAULT};
    MQIMPO   InqPropOpts       = {MQIMPO_DEFAULT};
    MQPD     PropDesc          = {MQPD_DEFAULT};
    MQLONG   Type              = MQTYPE_AS_SET;
    MQCHARV  TopStrProp        = {MQTOPICSTRING};
    MQCHARV  PubOptProp        = {MQPUBOPTIONS};
    MQLONG   DataLength        = 0;
    MQBYTE   buffer[256]       = "";
    MQLONG   buflen            = sizeof(buffer) - 1;
    MQLONG   messlen           = 0;
    char     TopStrBuf[256]    = "Initial value";
    int      i                 = 0;
```

Şekil 61. Bildirimler

Bildirimlerde kolayca gerçekleştirilmeyen kullanıma hazırlama işlemleri Şekil 62 sayfa 299içinde gösterilir. Vurgulanan değerler açıklama gerektiriyor.

SYSTEM.NDURABLE.MODEL.QUEUE
In this example, instead of using MQSUB to open a managed non-durable subscription, the model queue, SYSTEM.NDURABLE.MODEL.QUEUE, is used to create a temporary dynamic queue. Tanıtıcısı MQSUB' a iletilir. Kuyruk doğrudan açılarak, tüm ileti bağlamını kaydedebilir ve abonelik seçeneğini (MQSO_SET_CORREL_ID) ayarlamasınız.

MQGMO_CURRENT_VERSION
WebSphere MQ yapılarının büyük kısmının yürürlükteki sürümünü kullanmak önemlidir. gmo.MsgHandle gibi alanlar yalnızca denetim yapılarının en son sürümünde kullanılabilir.

MQGMO_PROPERTIES_IN_HANDLE
Özgün yayında konu dizgisi ve koyma iletisi seçenekleri, ileti özelliklerini kullanarak araya giriş abonesi tarafından alınmak üzere ayarlanır. Diğer bir seçenek, iletide **MQRFH2** yapısının doğrudan okunmasını sağlar.

MQSO_SET_CORREL_ID
Birlikte MQSO_SET_CORREL_ID ile birlikte kullanılması,

```
memcpy(sd.SubCorrelId, MQCI_NONE, sizeof(sd.SubCorrelId));
```

Bu seçeneklerin etkisi ilinti tanıtıcısından geçmektedir. Özgün yayınlayıcı tarafından ayarlanan ilinti tanıtıcısı, yayınlama abonesi tarafından alınan yayının ilinti tanıtıcısı alanına yerleştirilir. Her bir araya gelen abone aynı ilinti tanıtıcısında geçiyor. Son abonede, aynı ilinti tanıtıcısını alma seçeneği vardır.

Not: Yayınlama bir yayınlama/abone olma sıradüzeninden geçirilirse, ilinti tanıtıcısı hiçbir zaman alıkonmaz.

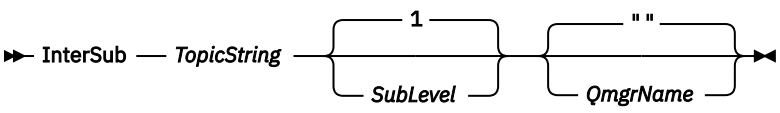
MQPRI_PRIORITY_AS_PUBLISHED
Yayın, yayınlandığı şekliyle aynı ileti önceliğine sahip yayın kuyruğuna yerleştirilir.

```
strncpy(getOD.ObjectName,    "SYSTEM.NDURABLE.MODEL.QUEUE",
        sizeof(getOD.ObjectName));
gmo.Version                = MQGMO_VERSION_4;
gmo.Options                = MQGMO_WAIT
                           | MQGMO_PROPERTIES_IN_HANDLE
                           | MQGMO_CONVERT;
gmo.WaitInterval           = 30000;
sd.Options                 = MQSO_CREATE
                           | MQSO_FAIL_IF_QUIESCING
                           | MQSO_SET_CORREL_ID;
sd.PubPriority             = MQPRI_PRIORITY_AS_PUBLISHED;
sd.Version                 = MQSD_VERSION_1;
memcpy(sd.SubCorrelId, MQCI_NONE, sizeof(sd.SubCorrelId));
putOD.ObjectType             = MQOT_TOPIC;
putOD.ObjectString.VSPtr     = &TopStrBuf;
putOD.ObjectString.VSBufSize = sizeof(TopStrBuf);
putOD.ObjectString.VSLength  = MQVS_NULL_TERMINATED;
putOD.ObjectString.VSCCSID   = MQCCSI_APPL;
putOD.Version                = MQOD_VERSION_4;
pmo.Version                = MQPMO_VERSION_3;
```

Şekil 62. Kullanıma Hazırlama

Şekil 63 sayfa 300 , komut satırı parametrelerini okuyacak, kullanıma hazırlama işlemini tamamlanacak ve kesişme aboneliğini oluşturabilmek için kod parçasını gösterir.

Programı komutla çalıştırın.



Hata işlenmesini mümkün olduğunca açık hale getirmek için, her bir MQI çağrısından neden kodu farklı bir dizi öğesinde saklanır. Her çağrıdan sonra tamamlanma kodu test edilir ve bu değer MQCC_FAIL ise, denetim do { } while(0) kod öbeğinden çıkar.

İki kayda değer kod satırı,

pmo.PubLevel = sd.SubLevel - 1;
Yeniden yayınlanan iletinin yayın düzeyini, kesilecek abonenin abonelik düzeyinden bir daha küçük bir düzeye ayarlar.

gmo.MsgHandle = Hmsg;
İleti özelliklerini döndürmek için MQGET için bir ileti tanıtıcısı sağlar.

```
do {
   printf("Intercepting subscriber start\n");
   if (argc < 2) {
       printf("Required parameter missing - topic string\n");
       exit(99);
   } else {
       sd.ObjectString.VSPtr    = argv[1];
       sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
       printf("TopicString = %s\n", sd.ObjectString.VSPtr);
   }
   if (argc > 2) {
       sd.SubLevel  = atoi(argv[2]);
       pmo.PubLevel = sd.SubLevel - 1;
       printf("SubLevel is %d, PubLevel is %d\n", sd.SubLevel, pmo.PubLevel);
   }
   if (argc > 3)
       strncpy(QMName, argv[3], sizeof(QMName));
   MQCONN(QMName, &Hcon, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   MQOPEN(Hcon, &getOD, GO_Options, &gHobj, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   MQSUB(Hcon, &sd, &gHobj, &Hsub, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   MQCRTMH(Hcon, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   gmo.MsgHandle = Hmsg;
```

Şekil 63. Yayınların kesişme hazırlığı yapılıyor

The main code fragment, Şekil 64 sayfa 301, gets messages from the publication queue. İleti özelliklerini sorgular ve konu dizesini kullanarak iletileri yeniden yayınlar ve yayının özgün **MQPMO**.seçeneği özelliklerini kullanır.

Bu örnekte, yayınında herhangi bir dönüşüm gerçekleştirilmez. Yeniden yayınlanan yayının konu dizgisi, her zaman, araya abone olunan abonenin abone olduğu konu dizgisiyle eşleşir. Kesişme abonesi, aynı yayın kuyruğuna gönderilen birden çok aboneliğin durdurulmasından sorumlu olursa, farklı aboneliklerle eşleşen yayınları ayırt etmek için konu dizesini sorgulamak gerekebilir.

MQINQMP çağrıları vurgulanır. Konu dizesi ve yayını, ileti seçenekleri özelliklerini doğrudan çıkış kontrol yapılarına yazılıp yazılır. putOD.ObjectString MQCHARV uzunluk alanını belirtik bir değerle boş olarak sonlandırılmış dizgiye değiştirmenin tek nedeni, dizginin çıkışını yapmak için printf kullanılmasıdır.

```
while (CompCode != MQCC_FAILED) {
   memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
   memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
   md.Encoding       = MQENC_NATIVE;
   md.CodedCharSetId = MQCCSI_Q_MGR;
   printf("MQGET : %d seconds wait time\n", gmo.WaitInterval/1000);
   MQGET(Hcon, gHobj, &md, &gmo, buflen, buffer, &messlen,
       &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   buffer[messlen] = '\0';
   MQINQMP(Hcon, Hmsg,    &InqPropOpts, &TopStrProp, &PropDesc, &Type,
       putOD.ObjectString.VSBufSize, putOD.ObjectString.VSPtr,
       &(putOD.ObjectString.VSLength), &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   memset((void *)((MQLONG)(putOD.ObjectString.VSPtr)
       + putOD.ObjectString.VSLength),'\0',1);
   putOD.ObjectString.VSLength = MQVS_NULL_TERMINATED;
   MQINQMP(Hcon, Hmsg, &InqPropOpts, &PubOptProp, &PropDesc, &Type,
       sizeof(pmo.Options), &(pmo.Options), &DataLength,
       &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   MQOPEN(Hcon, &putOD, PO_Options, &pHobj, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   printf("Republish message <%s> on topic <%s> with options %d\n",
       buffer, putOD.ObjectString.VSPtr, pmo.Options);
   MQPUT(Hcon, pHobj, &md, &pmo, messlen, buffer, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
       break;
   MQCLOSE(Hcon, &pHobj, PC_Options, &CompCode, &Reason);
   if (CompCode == MQCC_FAILED)
   break;
}
```

Şekil 64. Yayını engelle ve yeniden yayınla

Son kod parçası Şekil 65 sayfa 301içinde gösterilir.

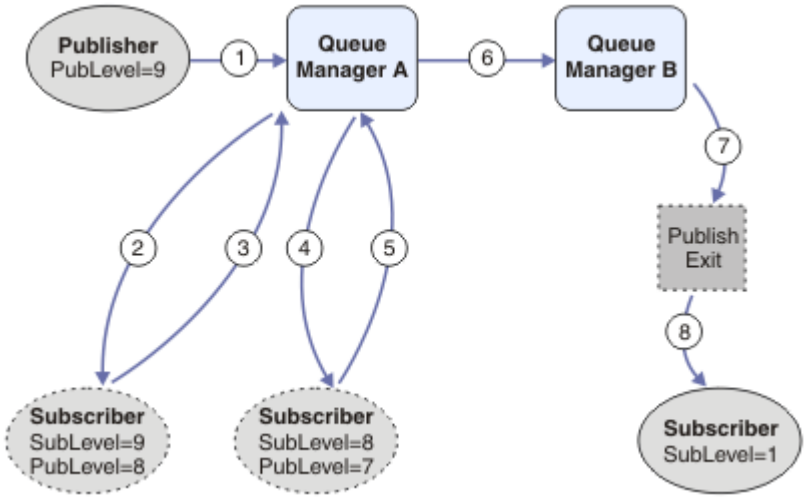
```
    } while (0);
    if (CompCode == MQCC_FAILED && Reason != MQRC_NO_MSG_AVAILABLE)
        printf("MQI Call failed with reason code %d\n", Reason);
    if (Hsub!= MQHO_NONE)
        MQCLOSE(Hcon, &Hsub, SC_Options, &CompCode, &Reason);
    if (Hcon!= MQHC_UNUSABLE_HCONN)
        MQDISC(&Hcon, &CompCode, &Reason);
}
```

Şekil 65. Tamamlanma

Yayınları ve dağıtımlı yayınlama/abone olma

Bir dağıtılmış yayınlama/abone olma topolojisinde araya girdiğinizde ya da yayınlama çıkışlarını konuşlandırdığınızda basit bir örüntüye uyun. Yayıncılarla aynı kuyruk yöneticilerindeki araya girmekte olan aboneleri konuşlandırın ve son aboneler olarak aynı kuyruk yöneticilerindeki çıkışları yayınlayın.

Şekil 66 sayfa 302 , bir yayınlama abone olma kümesine bağlı iki kuyruk yöneticisini gösterir. A publisher creates a publication to a cluster topic at publication level 9. Numaralandırılmış oklar, yayınla alınan adımların sırasını, abonelere küme konusuna akıştıkça gösterir. The publication is intercepted by the subscriber with `Alt düzey` 9 and republished with `Publevel` 8. `Alt düzey` 8' da bir abone tarafından yeniden yakalanır. Abone, `Publilevel` 7konumunda yeniden yayınlar. Kuyruk yöneticisi tarafından sağlanan yetkili sunucu abonesi, yayını kuyruk yöneticisi B ' ye iletir; burada son aboneye ek olarak bir Yayınlama çıkışı konuşlandırılmıştır. The publication is processed by the Publish exit before it is finally received by the final subscriber at `Alt düzey` 1. Araya gelen aboneler ve yayınlama çıkışı kırık çerçevelerle gösterilir.



Şekil 66. Bir kümedeki başlangıç ve yayınlama çıkışıdır

Yalın örüntünün amacı, aynı yayını almak için bir yayın alan her abone içindir. Bu yayın, abonenin bağlı olduğu yerden bağımsız olarak aynı dönüşümler sırasının üzerinden geçer. Yayıncıların ya da son abonelerin bağlı olduğu yere bağlı olarak, dönüşümlerin sırasının değişmesini önlemeniz gerekebilir. Makul bir kural dışı durum, yayını son olarak her bir aboneye teslim etmek için uyarlamak olacaktır. Yayının sonunda teslim edildiği kuyruğa dayalı olarak yayını özelleştirmek için Yayınlama çıkışını kullanın.

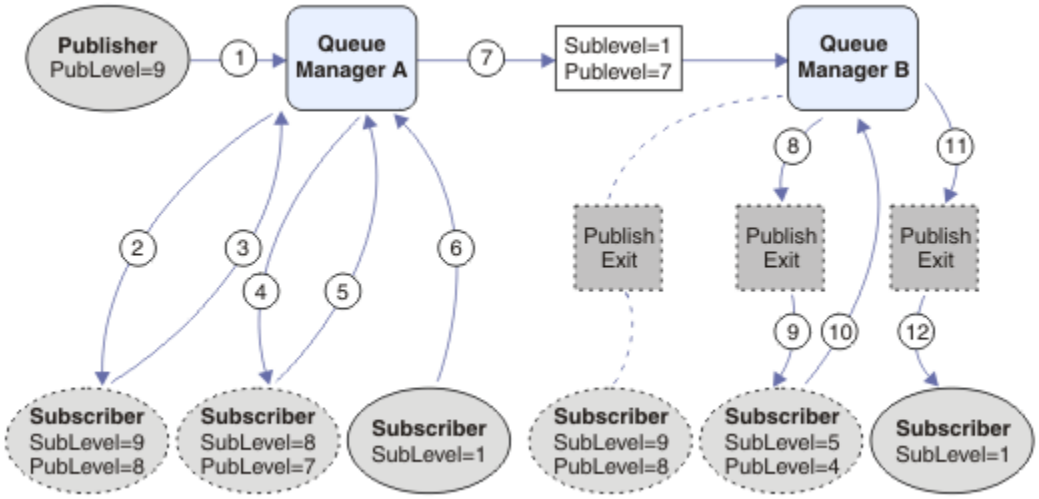
Dağıtılmış bir yayınlama/abone olma topolojisinde, kesişen abonelerin ve yayınlama çıkışlarının konuşlandırılacağı yeri dikkatli bir şekilde değerlendirmelisiniz. Bu düz örüntü, aboneleri yayıncılarla aynı kuyruk yöneticisine yerleştirir ve son abonelerle aynı kuyruk yöneticilerine yayınlar yayınlayın.

Düzenek karşıtı

Şekil 67 sayfa 303 shows how matters can go awry, if you do not follow a simple pattern. Konuşlandırmayı karmaşık hale getirebilmek için, kuyruk yöneticisine bir son abone eklenir ve iki ek kesici abonesi kuyruk yöneticisi B ' ye eklenir.

The publication is forwarded to queue manager B at `PubLevel` 7, where it is intercepted by a subscriber at `SubLevel` 5 before being consumed by the final subscriber at `SubLevel` 1. Yayınlama çıkışı, hem algılayıcı tüketiciye, hem de kuyruk yöneticisi B ' deki son tüketiciye iletilmeden önce yayını ele geçirmektedir. Bu yayın, Yayınlama çıkışı tarafından işlenmeden kuyruk yöneticisi A ' nın son abonesine ulaşır.

Bir yayınlama/abone olma topolojisinde, yetkili sunucu aboneleri `SubLevel` 1'a abone olur ve son arama abonesi tarafından ayarlanan `PubLevel` ' ı iletir. In Şekil 67 sayfa 303, the result is that the publication is not intercepted by the subscriber using `SubLevel` 9 at queue manager B.



Şekil 67. Kesişen abonelerin karmaşık devreye alınması

Yayın seçenekleri

İletilerin yayınlanma şeklini denetleyen çeşitli seçenekler vardır.

Yanıtlama yanıtı-abonelerden bilgi almak

If you do not want subscribers to be able to reply to publications they receive, it is possible to withhold information in the ReplyToQ and ReplyToQmgr fields of the MQMD by using the MQPMO_SUPPRESS_REPLYTO put-message option. Bu seçenek kullanılırsa, kuyruk yöneticisi bu bilgileri herhangi bir aboneye iletmeden önce yayın aldığında MQMD ' den kaldırır.

Bu seçenek, bir ReplyToQ ' u gerektiren bir rapor seçeneğiyle birlikte kullanılamaz; bu, çağrıya çağrılan MQRC_MISSING_REPLY_TO_Q ile başarısız olma girişiminde bulunursa kullanılamaz.

Yayın düzeyi

Yayın düzeylerinin kullanılması, hangi abonelerin yayını alacağını denetlemeye ilişkin bir yöntemdir. Yayın düzeyi, yayınla hedeflenen abonelik düzeyini belirtir. Yalnızca en yüksek abonelik düzeyine sahip olan abonelikler, yayının yayın düzeyinden daha az ya da bu yayın düzeyiyle aynı olan abonelikler için geçerli olan abonelikleri alır. Bu değer, sıfır ile dokuz aralığında olmalıdır; sıfır, en düşük yayın düzeyidir. Bu alanın ilk değeri 9 'tır. Yayın ve abonelik düzeylerinin kullanılarından biri de yayını engelle.

Yayının herhangi bir aboneye teslim edilmediği denetleniyor

Bir yayının abonelere teslim edilmediğini denetlemek için, MQPUT çağrısına sahip MQPMO_WARN_IF_NO_XX_ENCODE_CASE_ONE subs_matched put-message seçeneğini kullanın. Put işlemi tarafından MQCC_UYARI ve bir MQRC_NO_ALTCHATCHED bir neden kodu döndürülürse, yayın herhangi bir aboneliğe teslim edilemedi. Koyma işleminde MQPMO_RETAIN seçeneği belirtilirse, ileti alıkonur ve daha sonra tanımlanmış eşleşen abonelikle teslim edilir. Dağıtılmış bir yayınlama/abone olma sisteminde, MQRC_NO_ALTS_MATCHED neden kodu, yalnızca kuyruk yöneticisinde konu için kayıtlı yetkili sunucu aboneliği yoksa döndürülür.

Abonelik seçenekleri

İleti aboneliklerinin nasıl işleneceğini denetleyen birkaç seçenek vardır.

İleti kalıcılığı

Kuyruk yöneticileri, yayıncı tarafından ayarlanan abonelere ilettikleri yayınların sürekliliğini korurlar. Yayınlayıcı, kalıcılığı aşağıdaki seçeneklerden biri olacak şekilde ayarlar:

0
Kalıcı olmayan

1
Kalıcı

2
Kuyruk/konu tanımlaması olarak kalıcılık

Yayınlama/abone olma için yayınlayıcı, konu nesnesini ve **topicString** çözümlenen bir konu nesnesiyle çözülebilir. Yayınlayıcı, kuyruk/konu tanımlaması olarak kalıcılık belirtiyorsa, çözümlenen konu nesnesindeki varsayılan kalıcılık yayınlama için belirlenir.

Alıkonan yayınlar

Alıkonan yayınlar alındığında denetim yapmak için aboneler iki abonelik seçeneği kullanabilir:

Yalnızca istek üzerine yayınla, MQSO_PUBLICATIONS_ON_REQUEST

Bir abonenin yayınlarını aldığında denetime sahip olmasını istiyorsanız, MQSO_PUBLICATIONS_ON_REQUEST abonelik seçeneğini kullanabilirsiniz. Bir abone daha sonra MQSUBRQ çağrısını kullanarak (özgün MQSUB çağrısından döndürülen Hsub tanıtıcısını belirterek) bir konunun korunan yayınına gönderildiğini istemek için yayınları aldığında denetleyebilirler. MQSO_PUBLICATIONS_ON_REQUEST abonelik seçeneğini kullanan aboneler, alıkonmamış yayınlar almaz.

MQSO_PUBLICATIONS_ON_REQUEST değerini belirlerseniz, herhangi bir yayını almak için MQSUBRQ ' yı kullanmanız gerekir. MQSO_PUBLICATIONS_ON_REQUEST 'i kullanmayacaksa, iletiler yayınlandığı gibi iletilir.

Bir abone, MQSUBRQ çağrısını kullanıyorsa ve abonelikte genel arama karakterleri kullanıyorsa, abonelik, bir konu ağacındaki birden çok konu ya da düğüm ile eşleşebilir; tüm tutulan iletiler (varsa) aboneye gönderilecektir.

Bu seçenek, bir kuyruk yöneticisi, abone uygulaması çalışmasa bile, kullanıcı tarafından bir aboneye yayın göndermeye devam edeceği için, kalıcı aboneliklerde kullanıldığında özellikle yararlı olabilir. Bu, abone kuyruğunda iletilerin birikmesine neden olabilir. Bu oluşturma, abonenin MQSO_PUBLICATIONS_ON_REQUEST seçeneğini kullanarak kaydolması önlenebilir. Diğer bir seçenek olarak, istenmeyen iletilerin bir oluşturmasını önlemek için uygulamanıza uygun olması durumunda, dayanıklı olmayan abonelikler de kullanabilirsiniz.

Bir abonelik dayanıklıysa ve bir yayınlayıcı alıkonacaksa, abone uygulaması, yeniden başlatma işleminden sonra durum bilgilerini yenilemek için MQSUBRQ çağrısını kullanabilir. Daha sonra, abonenin MQSUBRQ çağrısını kullanarak durumunu düzenli olarak yenilemesi gerekir.

Bu seçeneği kullanarak MQSUB çağrısının sonucu olarak hiçbir yayın gönderilmez. Bağlantı kesildiğinde sürdürülen kalıcı abonelik, özgün abonelik bu seçeneği kullanacak şekilde yapılandırıldıysa, MQSO_PUBLICATIONS_ON_REQUEST seçeneğini kullanacaktır.

Yalnızca yeni yayınlar, MQSO_NEW_PATICATIONS_ONLY

Bir konuyla ilgili alıkonan bir yayın varsa, yayından sonra abonelik yapan aboneler bu yayının bir kopyasını alır. Bir abone, yapılmakta olan abonelikten daha önce yapılmış bir yayını almak istemezse, abone MQSO_NEW_XX_ENCODE_CASE_ONE publications_only abonelik seçeneğini kullanabilir.

Abonelikleri gruplama

Yayınları almak için bir kuyruk ayarladıysanız ve aynı kuyruğa yayınların beslenmesi için bir dizi çakışan abonelikler varsa, abonelikleri gruplamanızı göz önünde bulundurun. Bu durum, Çakışan abonelikleriçindeki örneğe benzer.

Bir konuya abone olduğunuzda, MQSO_GROUP_SUB seçeneğini ayarlayarak yinelenen yayınlar almaktan kaçınabilirsiniz. Sonuçta, gruptaki birden çok abonelik bir yayının konusuyla eşleştiğinde, yayının kuyruğa konmasından yalnızca bir abonelik sorumlu olur. Yayın konusu ile eşleşen diğer abonelikler yok sayılır.

Yayını kuyruğa yerleştirmekten sorumlu olan abonelik, herhangi bir genel arama karakteri ile karşılaşmadan önce en uzun eşleşen konu dizesine sahip olduğu temel alınarak seçilir. Bu, en yakın eşleşen abonelik olarak düşünülebiliyor. Özellikleri, MQSO_NOT_OWN_PUBS özelliğine sahip olup olmadığı da içinde olmak üzere, yayına yansıtılır. Bu durumda, eşleşen diğer abonelikler MQSO_NOT_OWN_PUBS özelliğine sahip olmasalar da, kuyruğa yayın gönderilmez.

Yinelenen yayınları ortadan kaldırmak için tüm aboneliklerinizi tek bir gruba yerleştiremezsiniz. Gruplanmış abonelikler aşağıdaki koşulları yerine getirmelidir:

1. Aboneliklerin hiçbiri yönetilmedi.
2. Bir abonelik grubu, yayınları aynı kuyruğa teslim eder.
3. Her abonelik aynı abonelik düzeyinde olmalıdır.
4. Gruptaki her abonelik için yayın iletisi, aynı ilinti tanıtıcısına sahiptir.

   Her abonelik sonucunun aynı ilinti tanıtıcısına sahip bir yayın iletisinde olmasını sağlamak için, MQSO_SET_COREL_ID ' u yayında kendi ilinti tanımlayıcınız yaratmak için ayarlayın ve her abonelikte **SubCorrelId** alanında aynı değeri ayarlayın. **SubCorrelId** değerini MQCI_NONEdeğerine ayarlamayın.

Ek bilgi için ../com.ibm.mq.ref.dev.doc/q100080_.dita#q100080_/mqso_group_sub konusuna bakın.

Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme

Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

Bunlar, bir kuyruk yöneticisinin bir nesneyi işlemesine neden olur. Her bir WebSphere MQ nesnesi tipinin öznitelikleri, Nesnelerin öznitelikleri alanında ayrıntılı olarak açıklanmıştır.

Bazı öznitelikler, nesne tanımlandığında ayarlanır ve yalnızca WebSphere MQ komutları kullanılarak değiştirilebilir; bu tür bir özniteliğe örnek olarak, bir kuyruğa konulan iletiler için varsayılan öncelik değeri verilebilir. Diğer öznitelikler kuyruk yöneticisinin çalışmasından etkilenir ve zaman içinde değişebilir; örnek, kuyruğun yürürlükteki derinliğini gösterir.

MQINQ çağrısını kullanarak çoğu özniteliğin yürürlükteki değerlerini sorgulayabilirsiniz. MQI, bazı kuyruk özniteliklerini değiştirebileceğiniz bir MQSET çağrısı da sağlar. Başka bir nesne tipinin özniteliklerini değiştirmek için MQI çağrılarını kullanamazsınız; bunun yerine şunu kullanmanız gerekir:

WebSphere MQ for Windows, UNIX ve Linux platformları için MQSC başvurusuiçinde açıklanan MQSC olanağı.

Not: Bu belgede, nesnelerin özniteliklerinin adları, bu belgede MQINQ ve MQSET çağrılarıyla kullandığınız biçimlerde gösterilir. Öznitelikleri tanımlamak, değiştirmek ya da görüntülemek için WebSphere MQ komutlarını kullandığınızda, konu bağlantılarındaki komutların açıklamalarında gösterilen anahtar sözcükleri kullanarak öznitelikleri tanımlamanız gerekir.

Hem MQINQ hem de MQSET çağrıları, tanınabilmek için seçicilerin dizilerini kullanırsorgulamak ya da ayarlamak istediğiniz öznitelikler. Birlikte çalışabileceğiniz her öznitelik için bir seçici vardır. Seçici adının bir öneki var, özniteliğin niteliği tarafından belirlenir:

MQCA_	Bu seçiciler, karakter verileri (örneğin, bir kuyruğun adı) içeren özniteliklere başvurur.
MQIA_	Bu seçiciler sayısal değerleri (örneğin, *CurrentQueueDepth*, kuyruklardaki ileti sayısı) ya da sabit bir değeri (queuegibi, kuyruk yöneticisinin eşitleme noktalarını destekleyip desteklemediği *SyncPoint*gibi) içeren özniteliklere başvurur.

MQINQ ya da MQSET çağrılarını kullanmadan önce, uygulamanızın kuyruk yöneticisine bağlı olması ve özniteliklerin belirlenmesi ya da aranması için nesneyi açmak üzere MQOPEN çağrısını kullanmanız gerekir. Bu işlemler, “Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198 ve “Nesnelerin açılması ve kapatılması” sayfa 206içinde açıklanmaktadır.

Nesne özniteliklerinin sorulması ve ayarlanmasıyla ilgili daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “Bir nesnenin özniteliklerinin sorulmasına neden oluyor” sayfa 306
- “MQINQ çağrısının başarısız olduğu bazı durumlar” sayfa 307
- “Kuyruk özniteliklerinin ayarlanması” sayfa 307

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

Bir nesnenin özniteliklerinin sorulmasına neden oluyor

Herhangi bir IBM WebSphere MQtürünün özniteliklerini sorgulamak için MQINQ çağrısını kullanın.

Bu aramaya giriş olarak, aşağıdaki bilgileri sağlamanız gerekir:

- Bağlantı tanıtıcısı.
- Bir nesne tanıtıcısı.
- Seçicilerin sayısı.
- Her bir seçici, MQCA_ * ya da MQIA_ * biçiminde olacak şekilde, öznitelik seçicileri dizisi. Her seçici, sorgulamak istediğiniz bir değeri olan bir özniteliği temsil eder ve her bir seçici, nesne tutamacının temsil ettiği nesne tipi için geçerli olmalıdır. Seçicileri herhangi bir siparişte belirtebilirsiniz.
- Sormakta olduğunuz tamsayı özniteliklerin sayısı. Tamsayı öznitelikleri sorulmayacaksa, sıfır değerini belirtin.
- The length of the character attributes buffer in *CharAttrLength*. Bu, her bir karakter özniteliği dizilimini tutmak için gereken uzunluklar toplamını en az bir toplamın olması gerekir. Karakter öznitelikleri sorulmayacaksa, sıfır değerini belirtin.

MQINQ ' un çıkışı şöyledir:

- Diziye kopyalanan bir tamsayı öznitelik değerleri kümesi. Değer sayısı *IntAttrCount*tarafından belirlenir. *IntAttrCount* ya da *SelectorCount* sıfırsa, bu parametre kullanılmaz.
- Karakter özniteliklerinin döndürülediği arabellek. Arabellek uzunluğu *CharAttrLength* parametresiyle verilir. *CharAttrLength* ya da *SelectorCount* sıfırsa, bu parametre kullanılmaz.
- Bir tamamlanma kodu. Tamamlanma kodu bir uyarı veriyorsa, arama yalnızca kısmen tamamlandı demektir. Bu durumda neden kodunu inceleyin.
- Bir neden kodu. Üç kısmi tamamlama durumu vardır:

– Seçici kuyruk tipi için geçerli değil
– Tamsayı öznitelikleri için yeterli alan yok
– Karakter öznitelikleri için yeterli alan yok

Bu durumların birden fazlası ortaya çıkarsa, geçerli olan ilk değer döndürülür.

Çıkış ya da sorgu için bir kuyruk açsanız ve bu, yerel olmayan bir küme kuyruğuna çözülürse, yalnızca kuyruk adı, kuyruk tipi ve ortak öznitelikleri sorgulayabilirsiniz. MQOO_BIND_ON_AçıK kullanıldıysa, ortak özniteliklerin değerleri seçilen kuyruklardır. MQOO_BIND_NOT_FIXED ya da MQOO_BIND_ON_GROUP kullanıldıysa ya da MQOO_BIND_AS_Q_DEF kullanılmadıysa ve *DefBind* kuyruk özniteliği MQBNDD_BIND_NOT_FIXED ise, değerler olası küme kuyruklarından oluşan rasgele bir değerindir. Ek bilgi için "MQOPEN ve kümeler" sayfa 331 ve MQOPEN başlıklı konuya bakın.

Not: Çağrı tarafından döndürülen değerler, seçilen özniteliklerin anlık görüntüleridir. Programınız döndürülen değerlerde işlem yapmadan önce öznitelikler değişebilir.

MQINQ' da MQINQ çağrısının bir açıklaması var.

MQINQ çağrısının başarısız olduğu bazı durumlar

Özniteliklerini sorgulamak için bir diğer ad açsanız, diğer ad kuyruğunun özniteliklerini (WebSphere MQ nesnesi başka bir kuyruğa erişmek için kullanılır), temel kuyruklardan değil, geri döndürmeniz gerekir.

Ancak, diğer adın çözdüğü temel kuyruğun tanımlaması kuyruk yöneticisi tarafından da açılır ve başka bir program, MQREAD ve MQINQ çağrılarınız arasındaki aralıktaki temel kuyruk kullanımını değiştirirse, MQINQ çağrısının başarısız olur ve MQRC_OBJECT_CHANGED neden kodunu döndürür. Diğer ad kuyruğu nesnesinin öznitelikleri değiştirilirse arama da başarısız olur.

Benzer bir şekilde, uzak bir kuyruğun özniteliklerini sorgulamak üzere açtığınızda, yalnızca uzak kuyruğa ilişkin yerel tanımlamanın özniteliklerini geri döndürmeniz gerekir.

Araştırığınız kuyruk öznitelikleri tipi için geçerli olmayan bir ya da daha çok seçici belirlerseniz, MQINQ çağrısı bir uyarıyla tamamlanır ve çıkışı aşağıdaki gibi ayarlar:

- Tamsayı öznitelikleri için, ilgili *IntAttrs* öğeleri MQIAV_NOT_UYGULANABILIR olarak ayarlanır.
- Karakter öznitelikleri için, *CharAttrs* dizgisinin karşılık gelen bölümleri yıldız işaretleri olarak ayarlanır.

Sormakta olduğunuz nesne özniteliklerinin tipi için geçerli olmayan bir ya da daha çok seçici belirlerseniz, MQINQ çağrısı başarısız olur ve MQRC_SELECTOR_ERROR neden kodunu döndürür.

Bir model kuyruğuna bakmak için MQINQ ' yi çağıramazsınız; MQSC olanağını ya da altyapınızda var olan komutları kullanın.

Kuyruk özniteliklerinin ayarlanması

MQSET çağrısını kullanarak kuyruk özniteliklerinin nasıl ayarlanacak bilgilerini öğrenmek için bu bilgileri kullanın.

MQSET çağrısını kullanarak yalnızca aşağıdaki kuyruk özniteliklerini ayarlayabilirsiniz:

- *InhibitGet* (uzak kuyruklar için değil)
- *DistList* (z/OSüzerinde değil)
- *InhibitPut*
- *TriggerControl*
- *TriggerType*
- *TriggerDepth*
- *TriggerMsgPriority*
- *TriggerData*

MQSET çağrısı, MQINQ çağrıyla aynı parametrelere sahip. Ancak, MQSET için, tamamlanma kodu ve neden kodu dışındaki tüm değiştirgeler giriş değiştirgeleridir. Kısmi tamamlama durumları yoktur.

Not: Yerel olarak tanımlanmış kuyruklar dışında, WebSphere MQ nesnelerinin özniteliklerini ayarlamak için MQI ' yı kullanamazsınız.

MQSET çağrısıyla ilgili daha fazla ayrıntı için bkz. MQSET.

İş birimlerinin kesinleştirilmesi ve yedeklenmesi

Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

Bu konuda aşağıdaki terimler kullanılmıştır:

- Kesinleştir
- İptal Et
- Syncpoint koordinasyonu
- Syncpoint
- İş birimi
- Tek aşamalı kesinleştirme
- İki aşamalı kesinleştirme

Bu işlem işleme terimlerini alıyorsanız, “IBM WebSphere MQ uygulamalarındaki eşitleme noktası konuları” sayfa 309' a atlayabilirsiniz.

Kesinleştir ve geri al

Bir program, bir iş birimi içindeki bir kuyruğa ileti yerleştirdiğinde, bu ileti yalnızca program iş birimini kesinleştirdiğinde diğer programlar tarafından görülebilir hale getirilmektedir. Bir iş birimini kesinleştirmek için, veri bütünlüğünü korumak için tüm güncellemelerin başarılı olması gerekir. Program bir hata saptarsa ve koyma işleminin kalıcı olmamaya karar verirse, iş birimini geri alabilirler. Bir program bir geri alma işlemi gerçekleştirdiğinde, IBM WebSphere MQ o iş birimi tarafından kuyruğa konulan iletileri kaldırarak kuyruğun geri depolarını geri yükler. Programın kesinleştirme ve geri çıkış işlemlerini gerçekleştirmesi, programın çalışmakta olduğu ortama bağlıdır.

Benzer bir şekilde, program bir iş birimi içindeki bir kuyruktan ileti aldığında, program iş birimini kesinleştirinceye kadar bu ileti kuyruktan kalır, ancak diğer programlar tarafından alınamayacak ileti görüntülenemez. Program, iş birimini kesinleştirdiğinde, ileti kuyruktan kalıcı olarak silinir. Program iş birimini yedeklerse, IBM WebSphere MQ , iletilerin diğer programlar tarafından alınabilmesini sağlanarak, kuyruğu geri yükler.

Syncpoint koordinasyonu, uyumluluk noktası, çalışma birimi

Syncpoint eşgüdümü , iş birimlerinin veri bütünlüğü ile kesinleştirileceği ya da yedekleneceği süreçtir.

Değişikliklerin kesinleştirme ya da geri alınması, en basit durumda, bir işlemin sonunda alınır. Ancak, bir uygulamanın veri değişikliklerini bir işlem içindeki diğer mantıksal noktalarda eşitlemesi daha kullanışlı olabilir. Bu mantıksal noktalar *eşitleme noktaları* (ya da *eşitleme noktaları*) olarak adlandırılır ve iki eşitleme noktası arasındaki bir güncelleme kümesinin işlenmesinin süresi *iş birimi*olarak adlandırılır. Birden çok MQGET çağrısı ve MQPUT çağrıları tek bir iş biriminin bir parçası olabilir. The maximum number of messages within a unit of work can be controlled by the MAXUMSGS attribute of the ALTER QMGR command on other platforms, except z/OS. Bu komutlara ilişkin ayrıntılar için MQSC başvurusu *WebSphere MQ Script (MQSC) Command Reference* adlı elkitabına bakın.

Tek aşamalı kesinleştirme

Bir *tek aşamalı kesinleştirme* işlemi, bir programın, değişikliklerini diğer kaynak yöneticileriyle koordine etmeden bir kuyrukta güncelleyebileceği bir işlemdir.

İki aşamalı kesinleştirme

Bir *iki aşamalı kesinleştirme* işlemi, bir programın IBM WebSphere MQ kuyruklarına yaptığı güncellemelerin diğer kaynaklarla (örneğin, DB2denetimi altındaki veritabanları) eşgüdümlü olarak eşgüdümlü olarak gerçekleştirilebileceğini gösteren bir işlemdir. Bu tür bir sürecin altında, tüm kaynaklarda yapılan güncellemeler kesinleştirilir ya da birlikte yedeklenir.

İş birimlerinin işlenmesine yardımcı olmak için IBM WebSphere MQ , *BackoutCount* özniteliğini sağlar. Bu, iş birimi içindeki bir iletinin geriletileceği her defasında artırılır. If the message repeatedly causes the unit of work to abnormally end, the value of the *BackoutCount* finally exceeds that of the *BackoutThreshold*. Bu değer, kuyruk tanımlandığında ayarlanır. Bu durumda uygulama, iletiyi çalışma biriminden kaldırabilir ve *BackoutRequeueQName*' ta tanımlandığı gibi başka bir kuyruğa yerleştirebilir. İleti taşındığında, iş birimi kesinleştirebilir.

İş birimlerinin kesinleştirilmesi ve yedeklenmesi hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “IBM WebSphere MQ uygulamalarındaki eşitleme noktası konuları” sayfa 309
- “UNIX, Linux, and Windows sistemlerinde IBM WebSphere MQ içindeki eşitleme noktaları” sayfa 310

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

IBM WebSphere MQ uygulamalarındaki eşitleme noktası konuları

IBM WebSphere MQ uygulamalarındaki eşitleme noktalarını kullanmaya ilişkin bilgi edinmek için bu bilgileri kullanın.

İki aşamalı kesinleştirme, aşağıda desteklenmektedir:

- WebSphere MQ for AIX
- HP-UXiçin WebSphere MQ
- Linuxiçin WebSphere MQ
- Solaris için WebSphere MQ
- WebSphere MQ for Windows
- MVS/ESA için CICS 4.1
- z/OSiçin CICS Transaction Server
- TXSeries
- IMS/ESA
- X/Open XA arabirimini kullanan diğer dış koordinatörler

Aşağıdakiler altında tek aşamalı kesinleştirme desteklenir:

- UNIX sistemlerinde WebSphere MQ

- WebSphere MQ for Windows

Not: Dış arabirimlerle ilgili ek ayrıntılar için bkz. “Dış eşitleme noktası yöneticilerine yönelik arabirimler” sayfa 312ve Open Group tarafından yayınlanan XA belgeleri *CAE Specification Distributed Transaction Processing: The XA Specification*. Hareket yöneticileri (CICS, IMS, Encinave Tuxedo gibi), diğer kurtarılabilir kaynaklarla koordine edilen iki aşamalı kesinleştirmeye katılabilirler. Başka bir deyişle, WebSphere MQ tarafından sağlanan kuyruğa alma işlevleri, hareket yöneticisi tarafından yönetilen bir iş birimi kapsamında getirilebilir.

WebSphere MQ ile verilen örnekler, XA uyumlu veritabanlarını koordine eden WebSphere MQ ile gösterilir. Bu örneklerle ilgili daha fazla bilgi için bkz. “Dağıtılmış platformlar için örnek programlar” sayfa 92.

WebSphere MQ uygulamanınızda, her put ve get çağrısına, çağrıyı syncpoint denetimi altında isteyip istemediğinizi belirleyebilirsiniz. Bir put işleminin syncpoint denetimi altında çalışması için, MQPUT ' u çağırdığınızda, MQPMO yapısının *Options* alanında MQPMO_SYNCPOINT değerini kullanın. Alma işlemi için, MQGMO yapısının *Options* alanında MQGMO_SYNCPOINT değerini kullanın. Bir seçeneği belirtik olarak seçmezseniz, varsayılan işlem altyapıya bağlıdır. Syncpoint denetimi varsayılan değeri hayır.

When an MQPUT1 call is issued with MQPMO_SYNCPOINT, the default behavior changes, so that the put operation is completed asynchronously. Bu, MQOD ve MQMD yapılarındaki bazı alanlara dayanan, ancak şimdi tanımsız değerler içeren bazı uygulamaların davranışlarında değişikliğe neden olabilir. Bir uygulama, koyma işleminin zamanuyumlu olarak gerçekleştirilmesini ve tüm uygun alan değerlerinin tamamlandığından emin olmak için MQPMO_SYNC_RESPONSE değerini belirtebilir.

Uygulamanız bir MQPUT ya da MQGET altında bir MQPUT ya da MQGET yanıtına yanıt olarak bir MQRC_BACKED_OUT neden kodu aldığında, uygulamanın olağan durumda MQBACK kullanarak yürürlükteki hareketi geri alması ve uygunsa, işlemin tamamını yeniden deneyin. Uygulama bir MQCMIT ya da MQDISC çağrısına yanıt olarak MQRC_BACKED_OUT alırsa, MQBACK ' ı çağırmasına gerek yoktur.

Bir MQGET çağrısı her yedeklendiğinde, etkilenen iletinin MQMD yapısındaki *BackoutCount*alanı artırılır. Yüksek *BackoutCount* , sürekli olarak yedeklenen bir iletiyi belirtir. Bu, araştırmanız gereken bu iletiyle ilgili bir sorun olduğunu gösterebilir. *BackoutCount*ile ilgili ayrıntılar için bkz. BackoutCount .

If a program issues the MQDISC call while there are uncommitted requests, an implicit syncpoint occurs. Program olağan dışı sona ererse, örtük bir geri alma gerçekleşir.

Kuyruk özniteliklerinde (MQSET çağrısıyla ya da komutlarla) yapılan değişiklikler, iş birimlerinin kesinleştirilmesiyle ya da yedeklenmesiyle etkilenmez.

UNIX, Linux, and Windows sistemlerinde IBM WebSphere MQ içindeki eşitleme noktaları

Syncpoint desteği iki tip iş birimi üzerinde çalışır: yerel ve küresel.

Yerel bir iş birimi, yalnızca WebSphere MQ kuyruk yöneticisinin güncellendiği tek kaynakların bir kaynaktır. Burada syncpoint eşgüdümü, kuyruk yöneticisinin tek aşamalı kesinleştirme yordamı kullanılarak sağlanıyor.

genel iş birimi, veritabanları gibi diğer kaynak yöneticilerine ait kaynakların da güncellendiği bir iş birimidir. WebSphere MQ , bu tür birimleri koordine edebilir. Bunlar, başka bir hareket yöneticisi ya da IBM i kesinleştirme denetleyicisi gibi bir dış kesinleştirme denetleyicisi tarafından da eşgüdümlü olarak kullanılabilir.

Tam bütünlük için, iki aşamalı kesinleştirme yordamı kullanın. İki aşamalı kesinleştirme, XA uyumlu hareket yöneticileri ve IBM' in TXSeries veUDBUDBgibi veritabanları tarafından sağlanabilir. WebSphere MQ ürünleri (IBM i için WebSphere MQ ve z/OSiçin WebSphere MQ dışında), iki aşamalı kesinleştirme işlemi kullanarak genel iş birimlerini koordine edebilir.

Yerel iş birimleri

Yalnızca kuyruk yöneticisini içeren iş birimleri *yerel* iş birimleri olarak adlandırılır. Syncpoint eşgüdümü, tek aşamalı kesinleştirme işlemi kullanılarak kuyruk yöneticisinin kendisi (iç koordinasyon) tarafından sağlanır.

Yerel bir iş birimi başlatmak için, uygulama MQGET, MQPUT ya da MQPUT1 istekleri uygun syncpoint seçeneğini belirtmektedir. İş birimi MQCMIT kullanılarak kesinleştirilir ya da MQBACK kullanılarak geriye işlendi. Ancak, iş birimi, uygulama ile kuyruk yöneticisi arasındaki bağlantı kesildiğinde, kasıtlı olarak ya da yanlışlıkla kesildiğinde de sona erer.

If an application disconnects (MQDISC) from a queue manager while a global unit of work coordinated by WebSphere MQ is still active, an attempt is made to commit the unit of work. Ancak, uygulama kesilmeden sona erdirilirse, uygulama olağandışı sona ermiş olduğu için, iş birimi geri alınır.

Genel iş birimleri

Diğer kaynak yöneticilerine ait kaynaklara ilişkin güncellemeleri de içermeniz gerektiğinde genel iş birimlerini kullanın.

Burada, eşgüdümleme, kuyruk yöneticisinin iç ya da dış dışkıyla birlikte olabilir:

İç eşitleme noktası eşgüdümü

Genel iş birimlerinin kuyruk yöneticisi eşgüdümü, IBM i için WebSphere MQ ya da z/OSiçin WebSphere MQ için desteklenmez. Bir WebSphere MQ MQI istemci ortamında desteklenmez.

Here, WebSphere MQ does the coordination. Genel bir iş birimi başlatmak için, uygulama MQSTART çağrısını yayınlar.

MQBEGIN çağrısına giriş olarak, MQCONN ya da MQCONNX çağrısının döndürdüğü bağlantı tanıtıcısını (*Hconn*) sağlamanız gerekir. Bu tanıtıcı, WebSphere MQ kuyruk yöneticiyle olan bağlantıyı gösterir.

Uygulama, uygun syncpoint seçeneğini belirterek MQGET, MQPUT ya da MQPUT1 isteklerine ilişkin sorunları yayınlar. Bu, yerel kaynakları, diğer kaynak yöneticilerine ait kaynakları ya da her ikisini de güncelleştiren genel bir iş birimi başlatmak için MQBEGIN komutunu kullanabilirsiniz. Diğer kaynak yöneticilerine ait kaynaklarda yapılan güncellemeler, o kaynak yöneticisinin API 'si kullanılarak yapılır. Ancak, diğer kuyruk yöneticilerine ait olan kuyrukları güncellemek için MQI ' yi kullanamazsınız. Daha fazla iş birimi (yerel ya da genel) başlatmadan önce MQCMIT ya da MQBACK komutunu verin.

Genel iş birimi MQCMIT kullanılarak kesinleştirilir; bu, iş biriminde yer alan tüm kaynak yöneticilerine ilişkin iki aşamalı kesinleştirmeyi başlatır. A two-phase commit process is used whereby resource managers (for example, XA-compliant database managers such as DB2, Oracle, and Sybase) are first all asked to prepare to commit. Sadece hepsi hazırlanırsa, kesinleştirilmek isteniyorsa. Herhangi bir kaynak yöneticisi söz veremeyeceğine işaret ederse, bunun yerine her biri dışarı çıkmanız istenir. Diğer bir seçenek olarak, tüm kaynak yöneticilerine ilişkin güncellemeleri geri almak için MQBACK ' ı kullanabilirsiniz.

Genel bir iş birimi hala etkin durumdayken bir uygulama bağlantısı kesilirse (MQDISC), iş birimi kesinleştirilir. Ancak, uygulama kesilmeden sona erdirilirse, uygulama olağandışı sona ermiş olduğu için, iş birimi geri alınır.

MQBEGIN çıkışı bir tamamlanma kodu ve neden kodudur.

Genel bir iş birimi başlatmak için MQBEGIN 'i kullandığınızda, kuyruk yöneticisiyle yapılandırılmış tüm dış kaynak yöneticileri içerilir. Ancak, arama bir iş birimi başlatır, ancak aşağıdaki durumlarda bir uyarıyla tamamlanır:

- Katılımcı kaynak yöneticisi yok (yani, kuyruk yöneticisiyle hiçbir kaynak yöneticisi yapılandırılmadı).

ya da

- Bir ya da daha çok kaynak yöneticisi kullanılamıyor.

Bu durumlarda, iş biriminin yalnızca iş birimi başlatıldığında kullanılabilir olan kaynak yöneticilerine yapılan güncellemeleri içermesi gerekir.

Kaynak yöneticilerinden biri güncellemelerini kesinleştiremezse, tüm kaynak yöneticilerine güncellemelerinin geri işlenmesi bildirilir ve MQCMIT bir uyarıyla tamamlanır. Olağan dışı durumlarda (genellikle, işletmen müdahalesi) bir MQCMIT çağrısı, bazı kaynak yöneticileri güncelleştirmelerini kesinleştirirse, ancak diğerleri geri gönderirse başarısız olabilir; işin *karma* bir sonuçla tamamlandığı

varsayılır. Bu tür oluşumların, kuyruk yöneticisinin hata günlüğünde tanısı konur, böylece çözüm işlemi giderilebilir.

Genel bir iş birimi MQCMIT, ilgili tüm kaynak yöneticileri güncellemelerini kesinleştirirse başarılı olur.

MQBEGIN çağrısının açıklaması için bkz. MQBEGIN.

Dış eşitleme noktası eşgüdümü

Bu durum, WebSphere MQ dışında bir Syncpoint eşgüdümcüsü seçildiyse; örneğin, CICS, Encinaya da Tuxedo için bu durum oluşur.

Bu durumda, UNIX and Linux sistemlerinde WebSphere MQ ve Windows için WebSphere MQ , ilgilerini eşitleme noktası eşgüdümcüyle birlikte çalışma biriminin sonucuna kaydeder; böylece, kesinleştirilmemiş alma ya da koyma işlemlerini gerektiği şekilde kesinleştirebilir ya da geri alabilirler. Dış eşitleme noktası eşgüdümcüsü, bir ya da iki aşamalı kesinleştirme protokollerinin sağlanıp sağlanmadığını belirler.

Bir dış eşgüdümcü kullandığınızda, MQCMIT, MQBACK ve MQBEGIN komutu yayınlanamaz. Bu işlevlere yapılan çağrılar, MQRC_ENVIRONMENT_ERROR neden koduyla başarısız olur.

Dışarıdan eşgüdümlü bir çalışma biriminin başlatıldığı yol, syncpoint eşgüdümcüsü tarafından sağlanan programlama arabirimine bağlıdır. Belirtik bir çağrı gerekli olabilir. Belirtik bir çağrı gerekiyorsa ve bir iş birimi başlatılmadığında MQPMO_SYNCPOINT seçeneğini belirten bir MQPUT çağrısı yayınladığınızda, MQRC_SYNCPOINT_NOT_AVAM tamamlanma kodu döndürülür.

İş biriminin kapsamı, syncpoint eşgüdümcüsü tarafından belirlenir. Uygulama ile kuyruk yöneticisi arasındaki bağlantının durumu, iş biriminin durumu değil, bir uygulama sorunu olan MQI çağrılarının başarısını ya da başarısızlığı etkiler. Örneğin, bir uygulama, etkin bir iş birimi sırasında bir kuyruk yöneticisine bağlantı kesip yeniden bağlayabilir ve aynı iş birimi içinde daha fazla MQGET ve MQPUT işlemleri gerçekleştirebilirler. Bu, beklemedeki bir bağlantı kesme işlemi olarak bilinir.

CICS' in XA yeteneklerini kullanmayı tercih etseniz de, CICS programlarında WebSphere MQ API çağrılarını kullanabilirsiniz. XA kullanmıyorsa, CICS atomik iş birimi içinde kuyruğa koyma ve ileti alır ve kuyruklar kuyruktan ileti alır. Bu yöntemi seçmenin bir nedeni, iş biriminin genel tutarlılığın sizin için önemli olmamasıdır.

İş birimlerinizin bütünlüğü sizin için önemliyse, XA ' yı kullanmanız gerekir. XA ' yı kullandığınızda, CICS , iş birimi içindeki tüm kaynakların birlikte güncellendiğinden emin olmak için iki aşamalı bir kesinleştirme protokolü kullanır.

İşlem desteğinin ayarlanmasıyla ilgili daha fazla bilgi için bkz. “Hareket desteği senaryoları” sayfa 39ve ayrıca TXSeries CICS belgeleri; örneğin, *TXSeries for Multiplatforms CICS Administration Guide for Open Systems*.

Dış eşitleme noktası yöneticilerine yönelik arabirimler

WebSphere MQ on UNIX and Linux systems, and WebSphere MQ for Windows support coordination of transactions by external syncpoint managers that use the X/Open XA interface.

Bazı XA hareket yöneticileri (TXSeries), her XA kaynak yöneticisinin adını sağladığından emin olun. Bu, XA anahtarı yapısındaki name adlı dizedir. UNIX, Linux ve Windows sistemlerinde WebSphere MQ için kaynak yöneticisi MQSeries_XA_RMI adını taşır. XA arabirimlerine ilişkin ayrıntılar için, Open Group (Açık Grup) tarafından yayınlanan XA belgelerine *CAE Specification Distributed Transaction Processing: The XA Specification*bakın.

Bir XA yapılandırmasında, WebSphere MQ on UNIX, Linuxve Windows sistemlerinde bir XA Resource Managerrolü yerine getirilmektedir. Bir XA syncpoint eşgüdümcüsü bir XA Kaynak Yöneticisi kümesini yönetebilir ve her iki Kaynak Yöneticisinde işlemlerin kesinlemesini ya da geri alma işlemlerini uyumlulaştırır. Bu, statik olarak kayıtlı bir kaynak yöneticisi için bu şekilde çalışır:

1. Bir uygulama, bir hareket başlatmak istediğini eşitleme noktası eşgüdümcü'ine bildirir.
2. Syncpoint eşgüdümcüsü, yürürlükteki işlemi bildirmek için tanıdığı kaynak yöneticilerine çağrı gönderir.

3. Uygulama sorunları, yürürlükteki işlemle ilişkilendirilmiş kaynak yöneticileri tarafından yönetilen kaynakları güncellemek için çağrılar.
4. Uygulama, syncpoint koordinatörünün hareketi kesinleştirmesini ya da geri yüklemesini ister.
5. syncpoint eşgüdümcüsü, her kaynak yöneticisine, işlemi istenen şekilde tamamlamak için iki aşamalı kesinleştirme protokolleri kullanarak çağırır.

XA belirtimi, her Resource Manager için *XA Anahtarı*adı verilen bir yapı sağlamasını gerektirir. This structure declares the capabilities of the Resource Manager, and the functions that are to be called by the syncpoint coordinator.

Bu yapının iki sürümü vardır:

MQRMIXASwitch	Durağan XA kaynak yönetimi
MQRMIXASwitchDynamic	Devingen XA kaynak yönetimi

Bu yapıyı içeren kitaplıkların listesi için bkz. “IBM WebSphere MQ XA anahtar yapısı” sayfa 67.

The method that must be used to link them to an XA syncpoint coordinator is defined by the coordinator; consult the documentation provided by that coordinator to determine how to enable WebSphere MQ to cooperate with your XA syncpoint coordinator.

Syncpoint eşgüdümcüsü tarafından herhangi bir *xa_open* çağrısında geçirilen *xa_info* yapısı, denetlenmek üzere olan kuyruk yöneticisinin adı olabilir. Bu, MQCONN ya da MQCONNX 'e geçirilen kuyruk yöneticisi adıyla aynı formu alır ve varsayılan kuyruk yöneticisi kullanılacaksa boş bırakılabilir. Ancak, TPM ve AXLIB iki parametre parametresini kullanabilirsiniz.

TPM, işlem yöneticisi adına (örneğin, CICS) WebSphere MQ ' ya belirtmenizi sağlar. AXLIB, XA AX giriş noktalarının bulunduğu hareket yöneticisinde gerçek kitaplık adını belirtmenizi sağlar.

Bu değiştirgelerden birini ya da varsayılan olmayan bir kuyruk yöneticisini kullanırsanız, QMNAME parametresini kullanarak kuyruk yöneticisi adını belirtmeniz gerekir. Daha fazla bilgi için bkz. xa_open dizesinin CHANNEL, TRPTYPE, CONNAME ve QMNAME parametreleri.

Kısıtlamalar

1. Paylaşılan bir Hconn (“Shared (thread independent) connections with MQCONNX” sayfa 203' ta açıklandığı gibi) ile genel iş birimlerinin kullanılmasına izin verilmez.
2. Windows sistemlerinde, XA anahtarında bildirilen tüm işlevler _cdecl işlevleri olarak bildirilir.
3. Bir dış eşitleme noktası eşgüdümcüsü, aynı anda yalnızca bir kuyruk yöneticisini yönetebilir. Bunun nedeni, eşgüdümcünün her kuyruk yöneticisinde etkili bir bağlantı olduğundan ve bu nedenle bir kerede tek bir bağlantıya izin verildiğine ilişkin kuralın da söz konusu olduğu içindir.

   Not: Not: JEE sunucusunda çalışan bir JMS istemcisi uygulaması (CLIENT JEE uygulaması) bu kısıtlamaya sahip değil; bu nedenle, tek bir JEE sunucu yönetimli bir hareket, aynı hareket içinde birden çok kuyruk yöneticisini koordine edebilir. Ancak, bağ tanımları kipinde çalışan bir JMS sunucusu uygulaması, bir kerede tek bir bağlantıya izin verilen kuralın hala geçerli olduğunu kabul eder.
4. Syncpoint eşgüdümcüsü kullanılarak çalıştırılan tüm uygulamalar, yalnızca bu kuyruk yöneticisine etkin bir şekilde bağlandıkları için, eşgüdümcünün yönettiği kuyruk yöneticisine bağlanabilir. Bir bağlantı tanıtıcısı almak için MQCONN ya da MQCONNX yayınlamalıdır ve çıkış işleminden önce MQDISC yayınlanmalıdır. Alternatively, they can use the exit UE014015 for TXSeries CICS.

Starting IBM WebSphere MQ applications using triggers

Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

Kuyruklara hizmet veren bazı WebSphere MQ uygulamaları sürekli olarak çalışır, bu nedenle her zaman kuyruklara gelen iletileri almak için kullanılabilir. Ancak, kuyruklara gelen ileti sayısı tahmin edilemez olduğunda bunu istemeyebilirsiniz. Bu durumda, alınacak herhangi bir ileti olmadığında da, uygulamalar sistem kaynaklarını tüketebilirler.

WebSphere MQ , alınabilmekte kullanılabilir iletiler olduğunda, uygulamanın otomatik olarak başlatılmasına olanak tanıyan bir olanak sağlar. Bu olanak, *tetikleme*olarak bilinir.

Kanalların tetiklenmesine ilişkin bilgi edinmek için Tetikleme kanalları başlıklı konuya bakın.

Tetikleme nedir?

Kuyruk yöneticisi, *tetikleme olayları*' u oluşturan bazı koşulları tanımlar.

Bir kuyruk için tetikleme etkinleştirilirse ve bir tetikleme olayı ortaya çıkarsa, kuyruk yöneticisi *tetikleyici ileti* adlı bir kuyruğa *başlatma kuyruğu*gönderir. Başlatma kuyruğunda tetikleme iletisinin varlığı, bir tetikleme olayının ortaya çıktığını gösterir.

Kuyruk yöneticisi tarafından oluşturulan tetikleme iletileri kalıcı değil. Bu, günlüğe kaydetme işlemini azaltır (performansı iyileştirmeye) ve yeniden başlatma sırasında yinelemeleri en aza indirerek yeniden başlatma süresini iyileştirir.

Başlatma kuyruğunu işleyen program *tetikleme-izleyici uygulaması*olarak adlandırılır ve tetikleme iletisinde bulunan bilgilere dayalı olarak tetikleme iletisini okuyup uygun işlemi yapmak için bu işlevi görmeniz gerekir. Genellikle bu işlem, tetikleme iletisini oluşturan kuyruğu işlemek için başka bir uygulamayı başlatmaya başlamanız gerekir. Kuyruk yöneticisinin bakış açısından, tetikleme izleme uygulamasıyla ilgili özel bir şey yoktur; bir kuyruktan ileti okuyan başka bir uygulamadır (başlangıç kuyruğu).

Bir kuyruk için tetikleme etkinleştirilirse, bu tetikle ilişkilendirilmiş bir *süreç tanımlaması nesnesi* yaratabilirsiniz. Bu nesne, tetikleme olayına neden olan iletiyi işleyen uygulamaya ilişkin bilgileri içerir. Süreç tanımlaması nesnesi yaratıldıysa, kuyruk yöneticisi bu bilgileri alır ve tetikleme izleme uygulaması tarafından kullanılmak üzere tetikleyici iletisine yerleştirir. Bir kuyrukla ilişkilendirilmiş süreç tanımlamasının adı, *ProcessName* yerel kuyruk özniteliği tarafından verilir. Her kuyruk farklı bir süreç tanımlaması belirtebilir ya da birden çok kuyruk aynı süreç tanımlamasını paylaşabilir.

Bir kanalın başlangıcını tetiklemek istiyorsanız, bir süreç tanımlaması nesnesi tanımlamanıza gerek yoktur. Bunun yerine iletim kuyruğu tanımlaması kullanılır.

Tetikleme, aşağıdaki ortamlarda çalışan WebSphere MQ istemcileri tarafından desteklenir:

- UNIX and Linux sistemleri
- Windows sistemleri

İstemci ortamında çalışan bir uygulama, bir tam WebSphere MQ ortamında çalışan bir uygulama ile aynıdır, ancak bu uygulama istemci kitaplıklarıyla ilişkilendirmenizi sağlar. Ancak, tetikleme izleme programı ve başlatılacak uygulamanın her ikisi de aynı ortamda olmalıdır.

Tetikleme şunları içerir:

Uygulama kuyruğu
Uygulama kuyruğu , tetikleme tetikleyicisi olduğunda ve koşullar karşılandığında tetikleme iletilerinin yazılmasını gerektiren bir yerel kuyruklardır.

Süreç tanımlaması
Uygulama kuyruğunda, uygulama kuyruğundan ileti alacak uygulamanın ayrıntılarını içeren bir *süreç tanımlaması nesnesi* ilişkilendirilebilir. (Özniteliklerin listesi için Süreç tanımlamalarına ilişkin öznitelikler konusuna bakın.)

Bir tetikleyicinin bir kanalı başlatmasını istiyorsanız, bir süreç tanımlaması nesnesi tanımlamanıza gerek olmadığını unutmayın.

İletim kuyruğu
Tetikleyicinin bir kanal başlatmasını istiyorsanız, iletim kuyruğuna gereksinim duyarsınız.

AIX, HP-UX, IBM i, Solaris, z/OSya da Windows sistemlerinde bir iletim kuyruğu için, iletim kuyruğunun *TriggerData* özniteliği başlatılacak kanalın adını belirtebilir. Bu işlem, tetikleme kanallarına ilişkin süreç tanımlamasını değiştirebilir, ancak yalnızca bir süreç tanımlaması yaratılmadığında kullanılır.

Tetikleme olayı

Tetikleme olayı , kuyruk yöneticisi tarafından tetikleme iletisinin oluşturulmasına neden olan bir olaydır. Bu, genellikle bir uygulama kuyruğuna gelen bir iletidir, ancak diğer zamanlarda da gerçekleşebilir (bkz. “Tetikleme olayına ilişkin koşullar” sayfa 319). WebSphere MQ has a range of options to allow you to control the conditions that cause a trigger event (see “Tetikleme olaylarını denetleme” sayfa 323).

Tetikleme iletisi

Kuyruk yöneticisi bir tetikleme olayını tanıdığında bir *tetikleme iletisi* yaratır (bkz. “Tetikleme olayına ilişkin koşullar” sayfa 319). Başlatılacak uygulamayla ilgili tetikleyici ileti bilgilerine kopyalanır. Bu bilgiler, uygulama kuyruğundan ve uygulama kuyruğuyla ilişkili süreç tanımlaması nesnesinden gelir. Tetikleme iletileri sabit bir biçime sahiptir (bkz. “Tetikleyici İletilerinin Biçimi” sayfa 329).

Başlatma kuyruğu

Kullanıma hazırlama kuyruğu , kuyruk yöneticisinin tetikleme iletilerini yerleştirdiği yerel bir kuyruğdur. Bir başlatma kuyruğunun diğer ad kuyruğu ya da model kuyruğu olamayacağı unutulmadır. Kuyruk yöneticisi birden çok kullanıma hazırlama kuyruğuna sahip olabilir ve her biri bir ya da daha çok uygulama kuyruklarıyla ilişkilendirilir. A shared queue, a local queue accessible by queue managers in a queue-sharing group, can be an initiation queue on WebSphere MQ for z/OS.

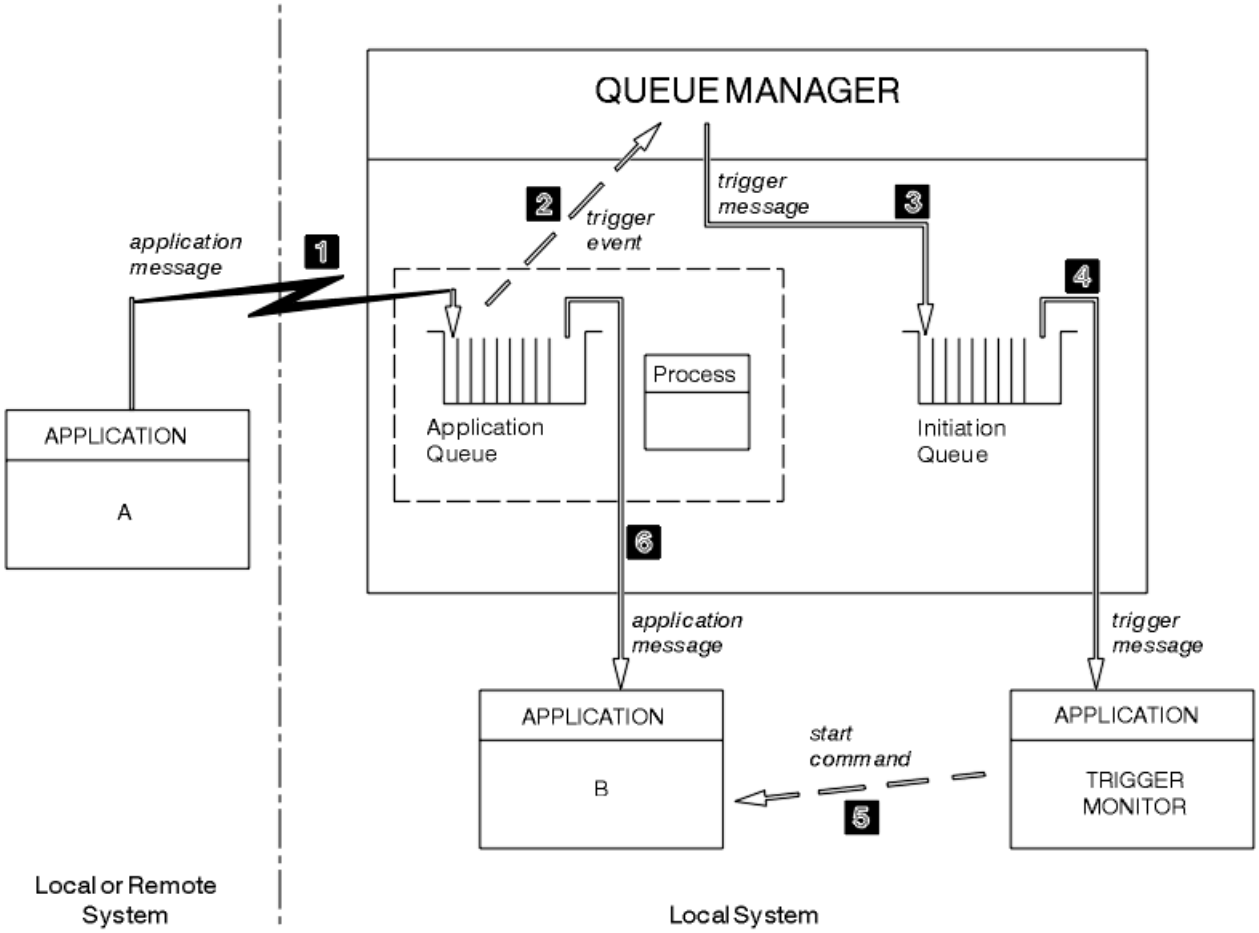
Tetikleyici İzleme Programı

Tetikleme izleme programı , bir ya da daha çok kullanıma hazırlama kuyruğuna hizmet veren sürekli çalışan bir programdır. Bir tetikleme iletisi bir başlatma kuyruğuna ulaştığında, tetikleme izleme programı iletiyi alır. Tetikleyici izleyicisi, tetikleme iletisinde yer alan bilgileri kullanır. Uygulama kuyruğuna gelen iletileri almak için, uygulama kuyruğunun adını içeren tetikleyici ileti üstbilgisinde yer alan bilgileri aktarmak için bir komut verir.

Tüm altyapılarda, kanal başlatıcı olarak bilinen özel bir tetikleme izleme programı, kanalların başlatılmasından sorumlu olur. z/OSüzerinde, kanal başlatıcısı manüel olarak başlatılır ya da kuyruk yöneticisi, kuyruk yöneticisi başlatma JCL 'de CSQINP2 ' i değiştirerek otomatik olarak başlatılabilmektedir. Diğer altyapılarda, kuyruk yöneticisi başlatıldığında otomatik olarak başlatılır ya da runmqchi komutu ile el ile başlatılabilir.

(Daha fazla bilgi için bkz. “Tetikleme izleyicileri tarafından kullanıma hazırlama kuyruğu işleme” sayfa 326.)

Tetikleme çalışmalarının nasıl çalıştığını anlamak için, FIRST (MQTT_FIRST) tetikleme tipinin bir örneği olan Şekil 68 sayfa 316' ı göz önünde bulundurun.



Şekil 68. Uygulama akışı ve tetikleme iletileri

Şekil 68 sayfa 316içinde olayların sırası şöyledir:

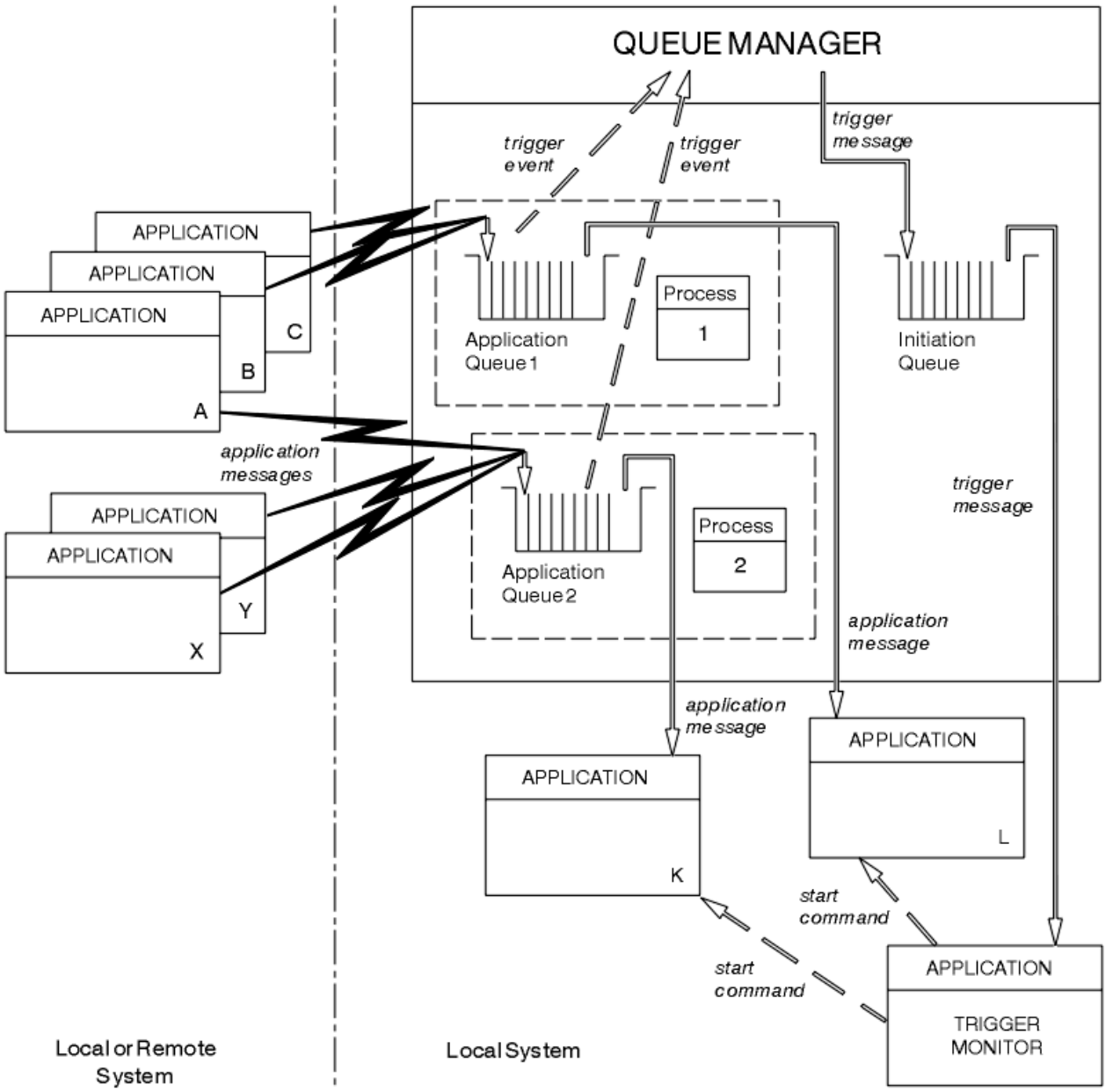
1. Yerel ya da kuyruk yöneticisinde uzak olabilen Uygulama A, bir iletiyi uygulama kuyruğuna yerleştirir. Bu kuyruk giriş için açık olan bir uygulama yok. Ancak, bu olgu yalnızca FIRST VE DEPTH tipini tetiklemek için geçerlidir.
2. Kuyruk yöneticisi, bir tetikleme olayı oluşturmak için sahip olduğu koşulların yerine getirilip karşılanmıyorsa, bunları görmek için denetler. Bunlar, bir tetikleme olayı oluşturulur. Tetikleme iletisi yaratılırken, ilişkili süreç tanımlaması nesnesi içinde tutulan bilgiler kullanılır.
3. Kuyruk yöneticisi bir tetikleme iletisi yaratır ve bu iletiyi bu uygulama kuyrukla ilişkilendirilmiş başlatma kuyruğuna koyar, ancak yalnızca bir uygulama (tetikleme izleme programı) giriş için açık kullanıma açma kuyruğu varsa, bu ileti kuyruğunu açar.
4. Tetikleyici izleyicisi, tetikleme iletisini başlatma kuyruğundan alır.
5. Tetikleme izleme programı, B uygulamasını (sunucu uygulaması) başlatmak için bir komut verir.
6. B uygulaması, uygulama kuyruğunu açar ve iletiyi alır.

Not:

1. Uygulama kuyruğu giriş için, herhangi bir program tarafından açılırsa ve FIRST ya da DEPTH için tetikleme tetiklemesi varsa, kuyruk zaten sunulmakta olduğu için tetikleme olayı oluşmaz.
2. Başlangıç kuyruğu giriş için açılmamışsa, kuyruk yöneticisi tetikleyici ileti üretmez; bir uygulama giriş için başlatma kuyruğunu açıncaya kadar bekler.
3. Kanallar için tetikleme kullanırken, FIRST ya da DEPTH tetikleyici tipini kullanın.

4. Tetikleme izleme programını başlatan kullanıcının kullanıcı kimliği ve grubu altında, CICS kullanıcısı ya da kuyruk yöneticisini başlatan kullanıcı tarafından çalıştırılmış uygulamalar.

Şu ana kadar, tetikleme içindeki kuyruklar arasındaki ilişki sadece tek bir temele dayandı. Bkz. Şekil 69 sayfa 317.



Şekil 69. Tetikleme içindeki kuyrukların ilişkisi

Uygulama kuyruğunda, iletiyi işleyecek uygulamanın ayrıntılarını içeren bir süreç tanımlaması nesnesi ilişkilendirilir. Kuyruk yöneticisi bilgileri tetikleme iletisine yerleştirir, bu nedenle yalnızca bir başlatma kuyruğu gereklidir. Tetikleme izleme programı bu bilgileri tetikleme iletisinden çıkarır ve her bir uygulama kuyruğunda iletiyle başa çıkmak için ilgili uygulamayı başlatır.

Bir kanalın başlangıcını tetiklemek istiyorsanız, bir süreç tanımlaması nesnesi tanımlamanıza gerek olmadığını unutmayın. İletim kuyruğu tanımlaması, tetiklenecek kanalı saptayabilir.

Tetikleyicileri kullanarak WebSphere MQ uygulamalarını başlatma hakkında daha fazla bilgi almak için aşağıdaki bağlantıları kullanın:

- “Tetikleme önkoşulları” sayfa 318

- “Tetikleme olayına ilişkin koşullar” sayfa 319
- “Tetikleme olaylarını denetleme” sayfa 323
- “Tetiklenen kuyrukları kullanan bir uygulama tasarlanması” sayfa 325
- “Tetikleme izleyicileri tarafından kullanıma hazırlama kuyruğu işleme” sayfa 326
- “Tetikleme İletilerinin Özellikleri” sayfa 329
- “Tetikleme işe yaramadığında” sayfa 330

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“MQI ve kümelerle çalışma” sayfa 330
Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

Tetikleme önkoşulları

Tetiklemeyi kullanmadan önce yapmanız gereken adımlar hakkında bilgi edinmek için bu bilgileri kullanın.

Uygulamanızın tetiklemeden yararlanabilmesi için aşağıdaki adımları tamamlayın:

1. Aşağıdakilerden birini yapın:
   a. Uygulama kuyruğunuz için bir başlatma kuyruğu yaratın. Örneğin:

```
DEFINE QLOCAL (initiation.queue) REPLACE      +
       LIKE (SYSTEM.DEFAULT.INITIATION.QUEUE)      +
       DESCR ('initiation queue description')
```

   ya da

   b. Var olan ve uygulamanız tarafından kullanılabilecek yerel bir kuyruğun adını saptayın (genellikle bu ad SYSTEM.DEFAULT.INITIATION.QUEUE YA DA, kanalları tetikleyicilerle başlatıyorsanız, SYSTEM.CHANNEL.INITQ) ve adını, uygulama kuyruğunun *InitiationQName* alanında belirtin.

2. Başlatma kuyruğunu uygulama kuyruğuyla ilişkilendirin. Kuyruk yöneticisi birden çok kullanıma hazırlama kuyruğuna sahip olabilir. Uygulama kuyruklarınızın bir kısmının farklı programlar tarafından sunulmasını isteyebilirsiniz. Bu durumda, her bir hizmet programı için bir başlatma kuyruğu kullanabilirsiniz, ancak buna gerek yoktur. Aşağıda, bir uygulama kuyruğu yaratılma örneği gösterilmektedir:

```
DEFINE QLOCAL (application.queue) REPLACE   +
```

```
    LIKE (SYSTEM.DEFAULT.LOCAL.QUEUE)           +
    DESCR ('appl queue description')    +
    INITQ ('initiation.queue')          +
    PROCESS ('process.name')            +
    TRIGGER                                     +
    TRIGTYPE (FIRST)
```

3. Bir uygulamayı tetikliyorsanız, uygulama kuyruğunuza hizmet verecek uygulamaya ilişkin bilgileri içerecek bir süreç tanımlaması nesnesi yaratın. Örneğin, PAYR adı verilen bir CICS bordro işlemini tetiklemek için:

```
    DEFINE PROCESS (process.name) +
           REPLACE +
           DESCR ('process description') +
           APPLICID ('PAYR') +
           APPLTYPE (CICS) +
           USERDATA ('Payroll data')
```

   Kuyruk yöneticisi bir tetikleme iletisi yarattığında, süreç tanımlaması nesnesinin özniteliklerinden alınan bilgileri tetikleyici iletisine kopyalar.

Altyapı	Süreç tanımlaması nesnesi yaratmak için
UNIX, Linuxve Windows sistemleri	İşLEM TANıMı YA DA SYSTEM.DEFAULT.PROCESS ve ALTER PROCESS kullanılarak değiştirme

4. İsteğe bağlı: Bir iletim kuyruğu tanımlaması yaratın ve *ProcessName* özniteliği için boşluk kullanın.

   TrigData özniteliği, tetiklenecek kanalın adını içerebilir ya da boş bırakılabilir. IBM WebSphere MQ for z/OSdışında, boş bırakılırsa, kanal başlatıcı, kanal tanımlama dosyalarını, adı belirtilen iletim kuyrukla ilişkili bir kanal buluncaya kadar arar. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, iletim kuyruğu tanımlamasının *TrigData* özniteliğinden tetikleme iletisine bilgi kopyalar.

5. Uygulama kuyruğunuza hizmet verecek uygulamanın özelliklerini belirlemek için bir süreç tanımlaması nesnesi yarattıktan sonra, süreç nesnesini, kuyruğun *ProcessName* özniteliğe adlayarak, uygulama kuyruğunuzla ilişkilendirin.

Altyapı	Komutları kullan
UNIX, Linuxve Windows sistemleri	ALTER QLOCAL

6. Tetikleyicinin eşgörünümlerini, tanımladığınız kullanıma hazırlama kuyruklarına sunmak için başlarını başlatın. Ek bilgi için “Tetikleme izleyicileri tarafından kullanıma hazırlama kuyruğu işleme” sayfa 326 başlıklı konuya bakın.

Teslim edilemeyen tetikleme iletilerinden haberdar olmak istiyorsanız, kuyruk yöneticinizin tanımlı bir ölü harf (teslim edilmemiş ileti) kuyruğu olduğundan emin olun. Specify the name of the queue in the *DeadLetterQName* queue manager field.

Daha sonra, gerek duyduğunuz tetikleme koşullarını, uygulama kuyruğunuzu tanımlayan kuyruk nesnesinin özniteliklerini kullanarak belirleyebilirsiniz. Daha fazla bilgi için “Tetikleme olaylarını denetleme” sayfa 323 başlıklı konuya bakın.

Tetikleme olayına ilişkin koşullar

Bu konuda paylaşılan kuyruklara yapılan başvurular, bir kuyruk paylaşım grubundaki paylaşılan kuyruklar anlamına gelir; yalnızca WebSphere MQ for z/OS' da kullanılabilir.

Kuyruk yöneticisi, aşağıdaki koşullar yerine getirildiğinde bir tetikleme iletisi yaratır:

1. Bir ileti, bir kuyruğa *yerleştirilir* .

2. İleti, kuyruğun eşik tetikleme önceliğine eşit ya da bu değere eşit bir öncelik içeriyor. Bu öncelik, *TriggerMsgPriority* yerel kuyruk öznitelikinde ayarlanır; sıfır olarak ayarlanmışsa, herhangi bir ileti nitelenmektedir.
3. The number of messages on the queue with priority greater than or equal to *TriggerMsgPriority* was previously, depending on *TriggerType*:
   - Sıfır (tetikleme tipi MQTT_FIRST için)
   - Herhangi bir sayı (tetikleme tipi MQTT_EVERY tetikleyicisi için)
   - *TriggerDepth* eksi 1 (MQTT_DEPTH tetikleme tipi için)

   Not:

   a. Paylaşılmayan yerel kuyruklar için, kuyruk yöneticisi, bir tetikleme olayına ilişkin koşulların var olup olmadığını değerlendirirken hem kesinleştirilmiş hem de kesinleştirilmemiş iletiler sayılır. Sonuç olarak, kuyruktaki iletiler kesinleştirilmediği için, bu uygulamanın alması gereken iletiler olmadığında bir uygulama başlatılmış olabilir. Bu durumda, uygulamanın iletilerin gelmesini beklemesi için uygun bir *WaitInterval*ile bekleme seçeneğini kullanmayı düşünün.

   b. Yerel paylaşılan kuyruklar için, kuyruk yöneticisi yalnızca kesinleştirilmiş iletileri sayar.
4. Birinci ya da DEPTH tipini tetiklemek için, hiçbir programın ileti kaldırmak için uygulama kuyruğu açılmadı (yani, *OpenInputCount* yerel kuyruk özniteliği sıfır).

   Not:

   a. Paylaşılan kuyruklar için, birden çok kuyruk yöneticisi bir kuyruğa karşı çalışan izleme programlarını tetiklediğinde özel koşullar uygulanır. Bu durumda, bir ya da daha çok kuyruk yöneticisi giriş paylaşılan girişi için açık olursa, diğer kuyruk yöneticilerindeki tetikleme ölçütleri *TriggerType* MQTT_FIRST ve *TriggerMsgPriority* sıfır olarak değerlendirilir. Tüm kuyruk yöneticileri giriş için kuyruğu kapattığında, tetikleme koşulları kuyruk tanımlamasında belirtilen koşullara geri çevrir.

      Bu koşulun etkilediği bir örnek, uygulama kuyruğu için çalışan bir tetikleme izleme programı olan birden çok kuyruk yöneticisi QM1, QM2ve QM3 ' dir. Bir ileti, tetikleme için gerekli koşulları yerine getirdiğinde, başlatma kuyruğunda bir tetikleyici iletisi oluşturulur. QM1 üzerindeki tetikleyici izleyicisi tetikleme iletisini alır ve bir uygulamayı tetikler. Tetiklenen uygulama, paylaşılan giriş için uygulama kuyruğunu açar. From this point on the trigger conditions for application queue A are evaluated as *TriggerType* MQTT_FIRST, and *TriggerMsgPriority* zero on queue managers QM2 and QM3, until QM1 closes the application queue.

   b. Paylaşılan kuyruklar için, bu koşul her kuyruk yöneticisi için geçerli olur. Yani, kuyruk yöneticisinin kuyruk yöneticisi tarafından kuyruğa alma işlemi içinqueuekuyruk yöneticisinin kuyruğun *OpenInputCount* kuyruk yöneticisi tarafından oluşturulacağı bir kuyruk için sıfır olması gerekir. Ancak, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi, MQOO_INPUT_EXCLUSIVE seçeneğini kullanarak kuyruk açıksa, kuyruk paylaşım grubundaki kuyruk yöneticilerinden herhangi biri tarafından bu kuyruk için herhangi bir tetikleme iletisi üretilmez.

      Tetiklenen uygulama, giriş için kuyruğu açtığında, tetikleme koşullarının nasıl değerlendirileceğini değiştirmek için bu değişikliği kullanın. Yalnızca tek bir tetikleme izleyicinin çalıştığı senaryolarda, diğer uygulamalar aynı etkiyi yapabilir, çünkü benzer şekilde giriş için uygulama kuyruğunu açarlar. Uygulama kuyruğunun, tetikleme izleme programı tarafından başlatılan bir uygulama tarafından mı, yoksa başka bir uygulama tarafından mı açıldığı önemli değildir; bu, tetikleme ölçütlerinde değişikliğe neden olan başka bir kuyruk yöneticisinde giriş için kuyruğun açık olduğu gerçidir.
5. On WebSphere MQ for z/OS, if the application queue is one with a *Usage* attribute of MQUS_NORMAL, get requests for it are not inhibited (that is, the *InhibitGet* queue attribute is MQQA_GET_ALLOWED). Ayrıca, tetiklenen uygulama kuyruğu, MQUS_XMITQ *Usage* özniteliğine sahip bir kuyruksa, istek alma işlemi yapılamaz.
6. Aşağıdakilerden birini yapın:
   - Kuyruğa ilişkin *ProcessName* yerel kuyruk özniteliği boş değil ve o özniteliğe ilişkin tanımlanan süreç tanımlaması nesnesi yaratıldı ya da

- Kuyruğa ilişkin *ProcessName* yerel kuyruk özniteliği boştur, ancak kuyruk bir iletim kuyruğudur. Süreç tanımlaması isteğe bağlı olduğu için, *TriggerData* özniteliği başlatılacak kanalın adını da içerebilir. Bu durumda, tetikleme iletisi aşağıdaki değerleri içeren öznitelikleri içerir:
  - *QName*: kuyruk adı
  - *ProcessName*: boşluklar
  - *TriggerData*: tetikleme verileri
  - *ApplType*: MQAT_UNKNOWN
  - *ApplId*: boşluklar
  - *EnvData*: boşluklar
  - *UserData*: boşluklar

7. Bir kullanıma hazırlama kuyruğu yaratıldı ve *InitiationQName* yerel kuyruk özniteisinde belirtildi. Ayrıca:
   - Alma istekleri başlatma kuyruğu için engellenmez (yani, *InhibitGet* kuyruk özniteliği MQQA_GET_ALLOWALIZD 'dir).
   - Put isteklerinin kullanıma hazırlama kuyruğu için engellenmemesi gerekir (yani, *InhibitPut* kuyruk özniteliğinin MQQA_PUT_ALLOWALIZD olması gerekir).
   - Başlatma kuyruğunun *Usage* özniteliğinin MQUS_NORMAL olması gerekir.
   - Dinamik kuyrukların desteklendiği ortamlarda, başlatma kuyruğu, mantıksal olarak silinmiş olarak işaretlenen dinamik bir kuyruk olmamalıdır.
8. Bir tetikleme izleyicinin şu anda iletileri kaldırmak için açma (initiation) kuyruğu açık (yani, *OpenInputCount* yerel kuyruk özniteliğinin sıfırdan büyük olması).
9. Uygulama kuyruğuna ilişkin tetikleme denetimi (*TriggerControl* yerel kuyruk özniteliği) MQTC_ON olarak ayarlandı. Bunu yapmak için, kuyruğunuzu tanımladığınızda *trigger* özniteliğini ayarlayın ya da ALTER QLOCAL komutunu kullanın.
10. Tetikleyici tipi (*TriggerType* yerel kuyruk özniteliği) MQTT_NONE değil.

    Tüm gerekli koşullar karşılanırsa ve tetikleme koşuluna neden olan ileti bir iş biriminin bir parçası olarak konulursa, iş birimi tamamlanıncaya kadar tetikleyici izleme uygulaması tarafından tetikleme iletisi, iş biriminin kesinleştirilip kesinleştirilmediği ya da MQTT_FIRST ya da MQTT_DEPTH tetikleme tipi için kullanılabilir duruma gelinceye kadar, tetikleme iletisi, alma işlemi için kullanılabilir duruma gelmeyebilir.
11. Kuyruğa uygun bir ileti, *TriggerType* MQTT_FIRST ya da MQTT_DEPTH ve kuyruk için bir ileti yerleştirilir:
    - Daha önce boş değil (MQTT_FIRST), ya da
    - *TriggerDepth* ya da daha fazla ileti (MQTT_DEPTH) vardı

    and conditions “2” sayfa 320 through “10” sayfa 321 (excluding “3” sayfa 320) are satisfied, if in the case of MQTT_FIRST a sufficient interval (*TriggerInterval* queue-manager attribute) has elapsed since the last trigger message was written for this queue.

    Bu, kuyruktaki tüm iletileri işlemeden önce biten bir kuyruk sunucusu için izin verilmesine olanak sağlar. Tetikleme aralığının amacı, oluşturulan yinelenen tetikleyici ileti sayısını azaltmaktadır.

    Not: Kuyruk yöneticisini durdurup yeniden başladıysanız, *TriggerInterval zamanlayıcı* sıfırlanır. İki tetikleyici ileti üretmenin mümkün olduğu küçük bir pencere vardır. Bu pencere, kuyruğun tetikleyici özniteliği bir iletiyle aynı zamanda etkinleştirilecek şekilde ayarlandığı ve kuyruk daha önce boş (MQTT_FIRST) ya da *TriggerDepth* ya da daha fazla ileti (MQTT_DEPTH) olan bir kuyruk olmadığı zaman var olur.
12. Bir kuyruğa hizmet veren tek uygulama, bir MQCLOSE çağrısı, *TriggerType* MQTT_FIRST ya da MQTT_DEPTH için ve en az:
    - Bir (MQTT_FIRST) ya da
    - *TriggerDepth* (MQTT_DEPTH)

Yeterli öncelik (koşul “2” sayfa 320) kuyruğuna ilişkin iletiler ve “6” sayfa 320 ile “10” sayfa 321 arasındaki koşullar da yerine getirilmektedir.

Bu, bir MQGET çağrısını içeren bir kuyruk sunucusuna izin vermek için, kuyruğu boş bulur ve bu nedenle sona erer; ancak, MQGET ile MQCLOSE çağrıları arasındaki aralıkla bir ya da daha fazla ileti gelir.

Not:

a. Uygulama kuyruğuna hizmet veren program tüm iletileri alamazsa, bu durum kapalı bir döngüye neden olabilir. Program kuyruğun her kapanışında, kuyruk yöneticisi tetikleme izleyicinin sunucu programını yeniden başlatmasına neden olan başka bir tetikleyici iletisi yaratır.

b. Uygulama kuyruğuna hizmet eden program, alma isteğini geri çekerse (ya da program sona erdirilmeden önce), aynı durumda, bu işlem kuyruğun kapatılıp kapatılmadan önce sona erdirilmesinden sonra da aynı olur. Ancak, program alma isteğini yedeklemeden önce kuyruğu kapar ve kuyruk boş değilse, herhangi bir tetikleyici iletisi yaratılanmaz.

c. Böyle bir döngüye engel olmak için, sürekli olarak yedeklenen iletileri saptamak için MQMD ' nin *BackoutCount* alanını kullanın. Daha fazla bilgi için “Yedeklenen iletiler” sayfa 36 başlıklı konuya bakın.

13. Aşağıdaki koşullar, MQSET ya da bir komut kullanılarak karşılanır:

a. • *TriggerControl* , MQTC_ON olarak değiştirilir ya da

• *TriggerControl* zaten MQTC_ON ve *TriggerType*, *TriggerMsgPriority*ya da *TriggerDepth* (ilgiliyse) değeri değiştirilirse, bu değer

ve en azından:

• Bir (MQTT_FIRST ya da MQTT_EVERY) ya da

• *TriggerDepth* (MQTT_DEPTH)

messages on the queue of sufficient priority (condition “2” sayfa 320), and conditions “4” sayfa 320 through “10” sayfa 321 (excluding “8” sayfa 321) are also satisfied.

Bir uygulamanın ya da işletmenin tetikleme ölçütlerinin değiştirilmesine izin vermesi, bir tetikleyicinin koşullarının ortaya çıkmasına ilişkin koşulların yerine getirilmesine izin verilmemesine neden olur.

b. Bir kullanıma hazırlama kuyruğunun *InhibitPut* kuyruk özniteliği MQQA_PUT_INHIBITED öğesinden MQQA_PUT_ALLOWALTED değerine değişir ve en az:

• Bir (MQTT_FIRST ya da MQTT_EVERY) ya da

• *TriggerDepth* (MQTT_DEPTH)

messages of sufficient priority (condition “2” sayfa 320) on any of the queues for which this is the initiation queue, and conditions “4” sayfa 320 through “10” sayfa 321 are also satisfied. (Koşulları karşılayan her kuyruk için bir tetikleyici ileti oluşturulur.)

Bu, başlatma kuyruğunda MQQA_PUT_INHIMATED koşulu nedeniyle tetikleme iletilerinin oluşturulmamasına izin vermek içindir, ancak bu koşul şimdi değiştirildi.

c. Bir uygulama kuyruğuna ilişkin *InhibitGet* kuyruk özniteliği MQQA_GET_INHIBITED ile MQQA_GET_ALLOWESTED olarak değişir ve en az:

• Bir (MQTT_FIRST ya da MQTT_EVERY) ya da

• *TriggerDepth* (MQTT_DEPTH)

messages of sufficient priority (condition “2” sayfa 320) on the queue, and conditions “4” sayfa 320 through “10” sayfa 321, excluding “5” sayfa 320, are also satisfied.

Bu, uygulamaların yalnızca uygulama kuyruğundan ileti alabildiği zaman tetiklenebilmesini sağlar.

d. Tetikleme izleme programı, bir başlatma kuyruğundan giriş için bir MQOPEN çağrısı yayınlar ve en az:

• Bir (MQTT_FIRST ya da MQTT_EVERY) ya da

- *TriggerDepth* (MQTT_DEPTH)

  Kullanıma hazırlama kuyruğu olduğu uygulama kuyruklarının herhangi birinde yeterli önceliğe (koşul “2” sayfa 320) ilişkin iletiler ve “4” sayfa 320 - “10” sayfa 321 (“8” sayfa 321dışında) arasındaki koşullar da karşılanır ve başka bir uygulama giriş için açma (initiation queue) için açılmaz (koşulları karşılayan her kuyruk için bir tetikleyici ileti oluşturulur).

  Bu, tetikleme izleme programı çalışmazken kuyruklara ulaşan iletilerin ve kuyruk yöneticisinin yeniden başlatılması ve iletilerin (kalıcı olmayan) kaybolması için izin verilmesine olanak tanır.

14. MSGDLVSQ doğru biçimde ayarlanmış. MSGDLVSQ=FIFO ' u ayarlıyorsanız, iletiler ilk olarak ilk giren ilk sırada kuyruğa teslim edilir. İletinin önceliği yok sayılır ve kuyruğun varsayılan önceliği iletiye atanmaktadır. *TriggerMsgPriority* , kuyruğun varsayılan önceliğine göre daha yüksek bir değere ayarlandıysa, hiçbir ileti tetiklenmez. *TriggerMsgPriority* , kuyruğun varsayılan önceliğine eşit ya da bu önceliğe eşit olarak ayarlanırsa, FIRST, EVERY ve DEPTH tipi için tetikleme gerçekleşir. Bu tiplerle ilgili bilgi için, “Tetikleme olaylarını denetleme” sayfa 323altındaki *TriggerType* alanının açıklamasına bakın.

    MSGDLVSQ=PRIORITY değeri ve ileti önceliği *TriggerMsgPriority* alanına eşitse ya da daha büyükse, tetikleyici olayı için yalnızca *sayı* değeri olur. Bu durumda, FIRST, EVERY ve DEPTH tipi için tetikleyici ortaya çıkar. As an example, if you put 100 messages of lower priority than the *TriggerMsgPriority*, the effective queue depth for triggering purposes is still zero. Daha sonra kuyruğa başka bir ileti koyarsanız, ancak bu kez öncelik *TriggerMsgPriority*' den büyük ya da bu değere eşit olduğunda, etkin kuyruk derinliği sıfırdan bire yükselir ve *TriggerType* FIRST için koşul karşılanır.

Not:

1. From step “12” sayfa 321 (where trigger messages are generated as a result of some event other than a message arriving on the application queue), the trigger message is not put as part of a unit of work. Ayrıca, *TriggerType* MQTT_EVERY ise ve uygulama kuyruğunda bir ya da daha çok ileti varsa, yalnızca bir tetikleyici iletisi oluşturulur.
2. WebSphere MQ , MQPUT sırasında bir ileti keserse, tüm kesimler kuyruğa başarıyla yerleştirilinceye kadar bir tetikleme olayı işlenmez. Ancak, ileti bölümleri kuyruksa, WebSphere MQ bunları tetikleme amacıyla tek tek ileti olarak değerlendirir. Örneğin, üç parçaya bölünen tek bir mantıksal ileti, ilk MQPUT ve kesimlendi olduğunda yalnızca bir tetikleyici olayının işlenmesine neden olur. However, each of the three segments causes their own trigger events to be processed as they are moved through the WebSphere MQ network.

Tetikleme olaylarını denetleme

Tetikleme olaylarını, uygulama kuyruğunuzu tanımlayan bazı öznitelikleri kullanarak denetlemenizi sağlar. Bu bilgiler, tetikleme tiplerinin kullanılmasına ilişkin örnekler de verir: EVERY, FIRST ve DEPTH.

Tetiklemeyi etkinleştirebilir ve devre dışı bırakabilirsiniz; bir tetikleme olayına doğru sayılan iletilerin sayısını ya da önceliğini seçebilirsiniz. Nesnelerin özniteliklerialanında bu özniteliklerin tam açıklaması vardır.

İlgili öznitelikler şunlardır:

TriggerControl
Bir uygulama kuyruğu için tetiklemeyi etkinleştirmek ve devre dışı bırakmak için bu özniteliği kullanın.

TriggerMsgPriority
Bir tetikleme olayına doğru sayılması için bir iletinin sahip olması gereken öncelik alt sınırı. Uygulama kuyruğuna *TriggerMsgPriority* değerinden küçük bir öncelik iletisi gönderilirse, kuyruk yöneticisi bir tetikleme iletisi yaratılıp yaratılmayacağını belirlediğinde iletiyi yoksayar. *TriggerMsgPriority* sıfır olarak ayarlandıysa, tüm iletiler bir tetikleme olayına doğru sayılır.

TriggerType
NONE (Yok) tetikleme tipine ek olarak (yalnızca *TriggerControl* 'un OFF' ye ayarlanması gibi tetiklemeyi geçersiz kılar), bir kuyruğun olayları tetiklemek üzere duyarlılığını ayarlamak için aşağıdaki tetikleyici tiplerini kullanabilirsiniz:

Her	Tetikleme olayı, uygulama kuyruğuna her ileti geldiğinde ortaya çıkar. Bir uygulamanın birden çok örneğinin başlatılmış olmasını istiyorsanız, bu tetikleyici tipini kullanın.
Birinci	Tetikleme olayı, yalnızca uygulama kuyruğunda bulunan ileti sayısı sıfırdan bire değişirse gerçekleşir. Bir hizmet programının bir kuyruğa ilk ileti geldiğinde başlatılmasını istiyorsanız bu tetikleyici tipini kullanın, işlenecek başka ileti kalmayıncaya kadar devam edin, daha sonra sona erdirin. Kuyruğu boş oluncaya kadar her zaman işlemeniz gerekir. Ayrıca bkz. “ÖNCE tetikleme tipi özel durumu” sayfa 325 .
Derinlik	Tetikleme olayı, yalnızca uygulama kuyruğunda ileti sayısı *TriggerDepth* özniteliğinin değerine ulaştığında oluşur. Bu tetikleme tipinin tipik bir kullanımı, bir istek kümesine verilen tüm yanıtlar alındığında bir program başlatmaya neden olur. **Derinliğe göre tetikleme:** Kuyruk yöneticisi, derinliği tetikleyerek tetiklemeyi (< xph> < pv>TriggerControl< /pv> < /xph> özniteliğini kullanarak) bir tetikleyici iletisi yarattıktan sonra tetiklemeyi devre dışı bırakır. Uygulamanızın, bu gerçekleştikten sonra kendisini tetiklemeyi yeniden etkinleştirmesi gerekir (MQSET çağrısını kullanarak). Tetiklemeyi devre dışı bırakma eylemi syncpoint denetimi altında değildir; bu nedenle, bir iş birimi yedeklenerek tetikleme yeniden etkinleştirilemiyor. Bir program tetikleme olayına neden olan bir put isteğini geri çekerse ya da program sona erdirilirse, MQSET çağrısını ya da ALTER QLOCAL komutunu kullanarak tetiklemeyi yeniden etkinleştirmeniz gerekir.

TriggerDepth
Bir tetikleme olayının derinlemesine tetiklenmesine neden olan bir kuyruğun ileti sayısı.

Bir kuyruk yöneticisinin tetikleme iletisi oluşturması için karşılanması gereken koşullar, “Tetikleme olayına ilişkin koşullar” sayfa 319 içinde açıklanmıştır.

EVERY tetikleme tipinin kullanımı örneği

Motor sigortası için istek üreten bir uygulama düşünün. Uygulama, her seferinde aynı yanıtı belirlemek üzere bir dizi sigorta şirketine istek iletileri gönderebilir. Bu yanıt kuyruğunda EVERY tipinde bir tetikleyici tanımlayabilir ve böylece her yanıt geldiğinde yanıt, yanıtı işlemek için sunucunun bir eşgörünümünü tetikleyebilir.

FIRST tetikleme tipini kullanma örneği

Her bir gün işyerine ilişkin ayrıntıları baş ofise iletmek için şube ofisleri olan bir kuruluş düşünün. Bunların hepsi aynı zamanda, çalışma gününün sonunda, ve baş ofisinde tüm şube ofislerinden gelen detayları işleyen bir uygulama var. Baş ofise gelen ilk ileti, bu uygulamayı başlatan bir tetikleme olayına neden olabilir. Bu uygulama, kuyruğunda daha fazla ileti kalmayıncaya kadar işlemeye devam eder.

Tetikleme tipi DEPTH ' in kullanımı örneği

uçuş rezervasyonunu doğrulamak için tek bir istek oluşturan bir seyahat acente uygulaması düşünün. bir otel odası için rezervasyon onaylamak, bir araba kiralamak ve bazı gezginler çekleri sipariş etmek. Uygulama bu öğeleri dört istek iletisine ayırabilir ve her birini ayrı bir hedefe gönderilebilir. Yanıtlama kuyruğunda (değeri 4 değerine ayarlanmış bir derinlik), yalnızca dört yanıt geldiğinde yeniden başlatılacak şekilde, yanıtlama kuyruğunda (değere 4 değerine ayarlanmış) bir tetikleyici ayarlanabilir.

Dört yanıttan önce yanıt kuyruğunda başka bir ileti (büyük olasılıkla farklı bir istekse) varırsa, istekte bulunan uygulama erken tetiklenir. Bunu önlemek için, bir isteğe birden çok yanıt toplamak için DERINLIK tetiklemesi kullanılırken, her istek için her zaman yeni bir yanıt kuyruğu kullanın.

ÖNCE tetikleme tipi özel durumu

FIRST tetikleyicisiyle, başka bir ileti geldiğinde uygulama kuyruğunda önceden bir ileti varsa, kuyruk yöneticisi tipik olarak başka bir tetikleyici iletisi yaratmaz.

Ancak, kuyruğa hizmet veren uygulama gerçekten kuyruğu açmayabilir (örneğin, uygulama sona erebilir, büyük olasılıkla bir sistem sorunu nedeniyle). Süreç tanımlaması nesnesine yanlış bir uygulama adı konulduysa, kuyruğa hizmet veren uygulama hiçbir iletiyi almayacaktır. Bu durumlarda, uygulama kuyruğuna başka bir ileti gelirse, bu iletiyi (ve kuyrukta bulunan diğer iletileri) işlemek için çalışan bir sunucu yoktur.

Bununla başa çıkmak için, kuyruk yöneticisi aşağıdaki durumlarda daha fazla tetikleme iletisi yaratır:

- Uygulama kuyruğuna başka bir ileti gelirse, ancak kuyruk yöneticisi o kuyruk için son tetikleme iletisini yarattığından bu yana önceden tanımlanmış bir zaman aralığı geçtiyse. Bu zaman aralığı, *TriggerInterval*kuyruk yöneticisi özniteisinde tanımlı. Varsayılan değeri 999 999 milisaniyedir.
- WebSphere MQ for z/OSüzerinde, açık bir kullanıma hazırlama kuyruğu adı olan uygulama kuyrukları düzenli olarak taranır. Son tetikleyici iletisinden bu yana *TRIGINT* milisaniyeler iletildiyse ve kuyruk bir tetikleme olayı için koşulları karşıladığında ve CURDEPTH sıfırdan büyükse, bir tetikleyici iletisi oluşturulur. Bu süreç, geri durdurma tetiklemesi olarak adlandırılır.

Uygulamanınızda kullanılacak tetikleme aralığı için bir değer belirlenirken aşağıdaki noktaları göz önünde bulundurun:

- *TriggerInterval* değerini düşük bir değere ayarladıysanız ve uygulama kuyruğuna hizmet veren bir uygulama yoksa, FIRST tetikleme tipi EVERY tetikleme tipi gibi davrabilir. Bu, iletilerin uygulama kuyruğuna konulmakta olduğu hıza bağlıdır; bu, diğer sistem etkinliklerine bağlı olabilir. Bunun nedeni, tetikleme aralığı çok küçükse, tetikleme tipi FIRST, EVERY değil, her ileti bir uygulama kuyruğuna her ileti konursa başka bir tetikleyici iletisi oluşturulur. (Tetikleme tipi sıfır olan FIRST ile tetikleme tipi, EVERY tetikleyicisinin eşdeğeridir.)
- On WebSphere MQ for z/OS if you set *TRIGINT* to a low value, and there is no application serving the trigger type FIRST application queue, backstop triggering will generate a trigger message each time the periodic scan of application queues that name open initiation queues takes place.
- Bir iş birimi yedeklendiyse (bkz. Tetikleme iletileri ve iş birimleri) ve tetikleme aralığı bir yüksek değere (ya da varsayılan değer) ayarlandıysa, iş birimi yedeklendiğinde bir tetikleyici iletisi oluşturulur. Ancak, tetikleme aralığını düşük bir değere ya da sıfır değerine ayarladıysanız (tetikleme tipi FIRST TO DEAD LIKE TRIGGER TYPE EVERY gibi) birçok tetikleyici iletisi oluşturulabilir. İş birimi yedeklendiyse, tüm tetikleyici iletiler hala kullanılabilir kılınmaya devam eder. Oluşturulan tetikleyici ileti sayısı, tetikleme aralığına bağlıdır. Tetikleme aralığı sıfır olarak ayarlandıysa, ileti sayısı üst sınırı oluşturulur.

Tetiklenen kuyrukları kullanan bir uygulama tasarlanması

Uygulamalarınız için nasıl ayarlanacak, denetleyeceğini ve tetiklemeyi gördünüz. Burada, uygulamanızı tasarlarken dikkate almanız gereken bazı ipuçları bulunur.

İletilerin ve çalışma birimlerinin tetiklenmesi

Bir iş biriminin parçası olmayan tetikleme olayları nedeniyle yaratılan tetikleme iletileri, herhangi bir iş biriminin dışında, başka hiçbir iletiye bağımlı olmadan, başlatma kuyruğuna yerleştirilir ve tetikleme izleme programı tarafından hemen geri alınabilmekte kullanılır.

Bir iş biriminin parçası olan tetikleme olayları nedeniyle yaratılan tetikleme iletileri, iş biriminin kesinleştirildiği ya da yedeklenip yedeklenmediği, UOW çözüldüğünde kullanıma hazırlama kuyruğunda kullanılabilir kılınmaktadır.

Kuyruk yöneticisi bir tetikleme iletisini bir başlatma kuyruğuna koyamazsa, bu ileti, ölü-mektup (teslim edilemeyen ileti) kuyruğuna konacak.

Not:

1. Kuyruk yöneticisi, bir tetikleme olayına ilişkin koşulların var olup olmadığını değerlendirirken hem kesinleştirilmiş hem de kesinleştirilmemiş iletiler sayılıyor.

Birinci ya da DERINLIK tipinde tetikleme işlemi ile, istenen koşullar karşılandığında bir tetikleme iletisi her zaman kullanılabilir olacak şekilde, iş birimi geriletilmiş olsa bile, tetikleme iletileri kullanılabilir duruma getirilmektedir. Örneğin, FIRST tetikleyicisi ile tetiklenen bir kuyruğa ilişkin iş birimi içinde bir put isteği düşünün. Bu, kuyruk yöneticisinin tetikleyici iletisi yaratmasına neden olur. Başka bir iş biriminden başka bir put isteği ortaya çıkarsa, bu durum başka bir tetikleme olayına neden olmaz; çünkü, uygulama kuyruğunda ileti sayısı artık bir tetikleme olayına ilişkin koşulları yerine getirmeyen birinden ikiden ikiye çevrmiştir. Şimdi ilk iş birimi geriletilirse, ancak ikincisi kesinleştirilirse, bir tetikleyici iletisi hala yaratılır.

Ancak bu, tetikleme iletilerinin bazen bir tetikleme olayına ilişkin koşullar *değil* yerine getirildiğinde tetikleneceği anlamına gelir. tetiklemeyi kullanan uygulamalar her zaman bu durumu ele almak için hazırlanmalıdır. It is recommended that you use the wait option with the MQGET call, setting the *WaitInterval* to a suitable value.

Yaratılan tetikleme iletileri her zaman kullanılabilir kılınsın, iş biriminin gerilettiği ya da kesinleştirilip kesinleştirilmemiş olması.

2. Yerel paylaşılan kuyruklar için (yani, bir kuyruk-paylaşım grubundaki paylaşılan kuyruklar), kuyruk yöneticisi yalnızca kesinleştirilmiş iletileri sayar.

Tetiklenen kuyruktan ileti alınması

Tetiklemeyi kullanan uygulamaları tasarladığınızda, bir program başlatma tetikleyicisi ile uygulama kuyruğunda kullanılabilir olan diğer iletiler arasında bir gecikme olabileceğini dikkate alınız. Tetikleme olayına neden olan ileti, diğerlerinden önce kesinleştirildiğinde bu durum oluşabilir.

İletilerin gelmesine izin vermek için, MQGET çağrısını kullanırken tetikleme koşullarının belirlendiği bir kuyruktan iletileri kaldırmak için her zaman bekleme seçeneğini kullanın. *WaitInterval* , iletilmekte olan bir ileti arasında en uzun makul süre için yeterli olmalıdır ve bu süre için çağrı kesinleştirilmelidir. İleti uzak bir kuyruk yöneticisinden geldiyse, bu süre aşağıdaki durumdan etkilenir:

- Kesinleştirilmeden önce konacağı iletilerin sayısı
- İletişim bağlantısının hızı ve kullanılabilirliği
- İletilerin büyüklükleri

MQGET çağrısını bekleme seçeneği ile kullanmanız gereken bir durum örneği için, iş birimlerini tanımlarken kullandığımız aynı örneği göz önünde bulundurun. Bu, FIRST tetikleyicisi ile tetiklenen bir kuyruk için çalışma birimi içindeki bir put isteğinde bulunmuyordu. Bu olay kuyruk yöneticisinin bir tetikleyici iletisi yaratmasına neden olur. Başka bir iş biriminden başka bir put isteği ortaya çıkarsa, bu, uygulama kuyruğunda ileti sayısı sıfırdan bire değişmediği için başka bir tetikleme olayına neden olmaz. Şimdi ilk iş birimi geriletilirse, ancak ikincisi kesinleştirilirse, bir tetikleyici iletisi hala yaratılır. Bu nedenle, tetikleme iletisi ilk iş biriminin yedekleneceği sırada yaratılır. İkinci iletinin kesinleştirilmesinden önce önemli bir gecikme süresi varsa, tetiklenen uygulamanın bunu beklemesi gerekebilir.

DERINLIK tipi tetiklenmesiyle, ilgili tüm iletiler sonunda kesinleştirilse de bir gecikme oluşabilir. *TriggerDepth* kuyruk özniteliğinin 2 değerine sahip olduğunu varsayın. Kuyruğa iki ileti geldiğinde, ikinci ileti bir tetikleme iletisinin yaratılmasına neden olur. Ancak, ikinci ileti kesinleştirilmek üzere ilk iletiyse, tetikleme iletisinin kullanılabilir duruma gelmesi o anda geçerli olur. Tetikleme izleme programı sunucu programını başlatır, ancak program ilk ileti kesinleştirilinceye kadar yalnızca ikinci iletiyi alabilir. Bu nedenle programın, ilk iletinin kullanıma sunulmasını beklemesi gerekebilir.

Uygulamanızı tasarlayın, böylece bekleme süreniz dolduğunda hiçbir ileti alınamazsa sona erdirilir. Bir ya da daha çok ileti daha sonra ulaşırsa, bunları işlemek için başvurunuzun yeniden denenmesine dikkat edin. Bu yöntem, uygulamaların boşta durmasını ve kaynakların gereksiz yere kullanılmasını önler.

Tetikleme izleyicileri tarafından kullanıma hazırlama kuyruğu işleme

Bir kuyruk yöneticisine, bir tetikleme izleme programı, kuyruğa hizmet veren diğer uygulamalar gibidir. Ancak, bir tetikleme izleme programı başlatma kuyruklarına hizmet eder.

Tetikleme izleme programı genellikle sürekli olarak çalışan bir programdır. Bir tetikleme iletisi bir başlatma kuyruğuna ulaştığında, tetikleme izleme programı o iletiyi alır. Bu ileti, uygulama kuyruğunda iletileri işlemek üzere olan uygulamayı başlatmak için bir komut vermek üzere iletiyle ilgili bilgileri kullanır.

Tetikleme izleme programının, doğru uygulama kuyruğunda doğru işlemleri gerçekleştirebilmesi için, programın başlatıldığı programa yeterli sayıda bilgi geçirmesi gerekir.

Kanal başlatıcı, ileti kanalı aracıları için özel bir tetikleyici izleyicisi örneğidir. Ancak bu durumda, FIRST YA DA DEPTH tetikleyicisinden birini kullanmanız gerekir.

UNIX ve Pencereler sistemlerinde tetikleme izleme programları

Bu konu, UNIX ve Windows sistemlerinde sağlanan tetikleme izleyicileri hakkındaki bilgileri içerir.

Sunucu ortamı için aşağıdaki tetikleme izleme programları sağlanmıştır:

amqstrg0
Bu, **runmqtrm**tarafından sağlanan işlevin bir alt kümesini sağlayan örnek bir tetikleme izleyicidir. amqstrg0ile ilgili ek bilgi için “Dağıtılmış platformlar için örnek programlar” sayfa 92 ' e bakın.

runmqtrm
Bu komutun sözdizimi şöyledir: **runmqtrm** [*-m QMgrName*] [*-q InitQ*]; burada QMgrName , kuyruk yöneticisi ve InitQ başlatma kuyruğunudur. Varsayılan kuyruk SYSTEM.DEFAULT.INITIATION.QUEUE varsayılan kuyruk yöneticisininde KUYRUK. Uygun tetikleyici iletilerine ilişkin programları çağırır. Bu tetikleyici izleyicisi, varsayılan uygulama tipini destekler.

Tetikleme izleyicinin işletim sistemine geçirilen komut dizilimi aşağıdaki gibi oluşturulur:

1. İlgili PROCESS (süreç) tanımlamasından *ApplId* (yaratıldıysa)
2. Çift tırnak işareti içine alınmış MQTMC2 yapısı
3. İlgili PROCESS (süreç) tanımlamasından *EnvData* (yaratıldıysa)

Burada *ApplId* , komut satırına girilirken çalıştırılacak programın adıdır.

İletilen parametre, MQTMC2 karakter yapısıdır. Sistem komutunun tek bir parametre olarak kabul etmesi için, tam olarak sağlandığı gibi, bu dizeye sahip olan bir komut dizgisi çağrılır.

Tetikleme izleme programı, başlatma kuyruğunda yeni başlatılmış olan uygulamanın tamamlanmasına kadar başka bir ileti olup olmadığını denetleyemez. Uygulamanın yapması gereken çok işlem varsa, tetikleme izleme programı gelen tetikleyici ileti sayısına yetişemeyebilir. İki seçeneğiniz vardır:

- Çalışan daha fazla tetikleyici izleme programı var
- Başlatılan uygulamaları arka planda çalıştır

Çalışmakta olan daha fazla tetikleyiciniz varsa, herhangi bir zamanda çalışabilecek uygulama sayısı üst sınırını denetleyebilirsiniz. Uygulamaları arka planda çalıştırırsanız, çalıştırılabilecek uygulama sayısında WebSphere MQ tarafından uygulanan bir kısıtlama yoktur.

To run the started application in the background on Pencereler systems, within the *ApplId* field, prefix the name of your application with a START command. Örneğin:

```
START ?B AMQSECHA
```

To run the started application in the background on UNIX systems, put an & at the end of the *EnvData* of the PROCESS definition.

Not: Bir Windows yolunun yol adının bir parçası olarak boşluklar varsa, bunlar tırnak işareti (") içine alınmalıdır. tek bir bağımsız değişken olarak ele alındığından emin olmak için. Örneğin, "C:\Program Files\Application Directory\Application.exe".

Aşağıda, dosya adının yolun bir parçası olarak boşluk bulunduğu bir APPLICID dizgisi örneği yer almaktadır:

```
START "" /B "C:\Program Files\Application Directory\Application.exe"
```

Örnekte, Windows START komutunun sözdizimi çift tırnak içine alınmış boş bir dizgi içerir. START, tırnak işaretlerindeki ilk bağımsız değişkenin yeni komutun başlığı olarak işleneceğini belirtir. Windows 'un' title ' bağımsız değişkenine ilişkin uygulama yolunu hata etmediğinden emin olmak için, uygulama adından önce komutta çift tırnak içine alınmış bir başlık dizgisi ekleyin.

Aşağıdaki tetikleme izleme programları WebSphere MQ istemcisi için sağlanır:

runmqtmc
This is the same as runmqtrm except that it links with the WebSphere MQ MQI client libraries.

CICSiçin
amqltmc0 tetikleyicisi izleyicisi CICSiçin sağlanmıştır. It works in the same way as the standard trigger monitor, runmqtrm, but you run it in a different way and it triggers CICS transactions.

Bu konu yalnızca Windows, UNIXve Linux sistemleri için geçerlidir.

Bu program bir CICS programı olarak sağlanır; bunu 4 karakterlik bir hareket adıyla tanımlayın. Tetikleme izleyicisini başlatmak için 4 karakterlik bir ad girin. Varsayılan kuyruk yöneticisini (qm.ini dosyasında adı ya da WebSphere MQ for Pencereler, kayıt dosyası) ve SYSTEM.CICS.INITIATION.QUEUE' yi kullanır.

If you want to use a different queue manager or queue, build the trigger monitor MQTMC2 structure: this requires you to write a program using the EXEC CICS START call, because the structure is too long to add as a parameter. Then, pass the MQTMC2 structure as data to the START request for the trigger monitor.

MQTMC2 yapısını kullandığınızda, diğer alanlara gönderme yapmadığı için tetikleyici izleyiciye yalnızca *StrucId*, *Version*, *QName*ve *QMgrName* parametrelerini sağlamanız gerekir.

İletiler, başlatma kuyruğundan okunur ve tetikleyici iletisinde APPL_TYPE 'ın MQAT_CICS olduğu varsayılarak, EXEC CICS START kullanılarak CICS işlemlerini başlatmak için kullanılır. İleti başlatma kuyruğundan gelen iletilerin okunması CICS syncpoint denetimi altında gerçekleştirilir.

İletiler, izleme programı başladığında ve durduğunda ve bir hata ortaya çıktığında oluşturulur. Bu iletiler CSMT geçici veri kuyruğuna yollanır.

Tetikleme izleyicisinin kullanılabilir sürümleri şunlardır:

S\u00fcr\u00fcm	**Kullan**
amqltmc0	TXSeries for AIX, HP-UX, and Sun Solaris Sürüm 5.1
amqltmc4	TXSeries for Windows, Sürüm 5.1
amqltmcc	CICS tetikleyicisi izleyicisinin istemci tanımlı sürümü

Başka ortamlar için bir tetikleme izleyicisine gereksinim duyarsanız, kuyruk yöneticisinin başlatma kuyruklarına koyduğu tetikleme iletilerini işleyebilecek bir program yazın. Böyle bir program aşağıdaki eylemleri gerçekleştirmelidir:

1. Bir iletinin başlatma kuyruğuna varmasını beklemek için MQGET çağrısını kullanın.
2. Başlatılacak uygulamanın adını bulmak için, tetikleme iletisinin MQTM yapısındaki alanları ve çalıştığı ortamı inceleyin.
3. Ortama özgü bir başlatma komutu verin.
4. MQTM yapısını gerekirse, MQTMC2 yapısına dönüştürün.
5. Başlatılan uygulamaya MQTMC2 ya da MQTM yapısını geçirin. Bu, kullanıcı verilerini içerebilir.

6. Uygulama kuyruğunuzla ilişkilendirin, o kuyruğa hizmet verecek olan uygulamayı ilişkilendirin. Bunu, kuyruğun *ProcessName* özniteisinde (yaratıldıysa) süreç tanımlaması nesnesini (yaratıldıysa) adlandırmayla yapabilirsiniz.

Tetikleyici izleme arabirimine ilişkin ek bilgi için MQTMC2başlıklı konuya bakın.

Tetikleme İletilerinin Özellikleri

Aşağıdaki konularda, ileti tetikleme iletilerinin diğer bazı özellikleri açıklanmaktadır.

- “Tetikleme iletilerinin sürekliliği ve önceliği” sayfa 329
- “Kuyruk yöneticisi yeniden başlatma ve ileti tetikleme” sayfa 329
- “İletilerin tetiklenmesi ve nesne özniteliklerinde yapılan değişiklikler” sayfa 329
- “Tetikleyici İletilerinin Biçimi” sayfa 329

Tetikleme iletilerinin sürekliliği ve önceliği

Tetikleme iletileri kalıcı değildir; bunun için gerekli bir gereksinim yoktur.

Ancak, tetikleme olaylarını oluşturma koşulları kalıcı olur; bu nedenle, bu koşullar karşılandığında tetikleme iletileri oluşturulur. Bir tetikleme iletisi kaybolursa, uygulama kuyruğunda uygulama iletisinin devam etmesi, tüm koşullar karşılandığı anda kuyruk yöneticisinin bir tetikleme iletisi oluşturmasını sağlar.

Bir iş birimi geriye işlenirse, üretilen tüm tetikleme iletileri her zaman teslim edilir.

Tetikleme iletileri, başlatma kuyruğunun varsayılan önceliğini alır.

Kuyruk yöneticisi yeniden başlatma ve ileti tetikleme

Bir kuyruk yöneticisinin yeniden başlatıldığı sırada, giriş için bir başlatma kuyruğu açıldığında, ilişkili bir uygulama kuyruğu üzerinde ileti varsa ve tetikleme için tanımlandıysa, bir tetikleme iletisi bu başlatma kuyruğuna konabilir.

İletilerin tetiklenmesi ve nesne özniteliklerinde yapılan değişiklikler

Tetikleme iletileri, tetikleme olayı sırasında zorlamalı olarak tetikleme özniteliklerinin değerlerine göre yaratılır.

Tetikleme iletisi tetikleme izleme programı tarafından daha sonra kullanılabilir kılınmadıysa (yaratılmasına neden olan ileti bir iş birimi içine konduysa), bu sırada tetikleme özniteliklerinde yapılan değişiklikler tetikleme iletisinde hiçbir etkiye sahip olmaz. Özellikle, tetiklemenin geçersiz kılınması, bir tetikleme iletisinin yaratıldıktan sonra kullanılabilir kılınmasını engellememektedir. Ayrıca, uygulama kuyruğu, tetikleme iletisinin kullanılabilir kılındığı zaman artık var olmayabilir.

Tetikleyici İletilerinin Biçimi

Bir tetikleme iletisinin biçimi, MQTM yapısı tarafından tanımlanır.

Bu, kuyruk yöneticisinin tetikleme iletisini yarattığında, uygulama kuyruğunun nesne tanımlamalarındaki ve o kuyrukla ilişkili sürecin nesne tanımlamalarındaki bilgileri kullanarak doldurduğu aşağıdaki alanları içerir:

StrucId
Yapı tanıtıcısı.

Version
Yapının sürümü.

QName
Tetikleme olayının ortaya çıktığı uygulama kuyruğunun adı. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, bu alanı uygulama kuyruğunun *QName* özniteliğini kullanarak doldurur.

ProcessName
Uygulama kuyruğuyla ilişkili süreç tanımlaması nesnesinin adı. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, bu alanı uygulama kuyruğunun *ProcessName* özniteliğini kullanarak doldurur.

TriggerData
Tetikleyici izleme programı tarafından kullanılmak üzere serbest biçimli bir alan. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, bu alanı uygulama kuyruğunun *TriggerData* özniteliğini kullanarak doldurur. Herhangi bir WebSphere MQ ürününde, WebSphere MQ for z/OSdışında, bu alan, tetiklenecek kanalın adını belirtmek için kullanılabilir.

ApplType
Tetikleme izleyicisinin başlatılacağı uygulamanın tipi. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, *ProcessName*içinde tanımlanan süreç tanımlaması nesnesinin *ApplType* özniteliğini kullanarak bu alanı doldurur.

ApplId
Tetikleme izleyicinin başlatılacağı uygulamayı tanıtan bir karakter dizilimi. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, *ProcessName*içinde tanımlanan süreç tanımlaması nesnesinin *ApplId* özniteliğini kullanarak bu alanı doldurur. When you use trigger monitor CKTI or CSQQTRMN supplied by WebSphere MQ for z/OS, the *ApplId* attribute of the process definition object is a CICS or IMS transaction identifier.

EnvData
Tetikleme izleme programı tarafından kullanılmak üzere ortama ilişkin verileri içeren bir karakter alanı. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, *ProcessName*içinde tanımlanan süreç tanımlaması nesnesinin *EnvData* özniteliğini kullanarak bu alanı doldurur. WebSphere MQ for z/OStarafından sağlanan tetikleme izleyicileri (CKTI ya da CSQQTRMN) bu alanı kullanmaz, ancak diğer tetikleme izleme programları bunu kullanmayı seçebilir.

UserData
Tetikleyici izleme programı tarafından kullanılmak üzere kullanıcı verilerini içeren bir karakter alanı. Kuyruk yöneticisi bir tetikleme iletisi yarattığında, *ProcessName*içinde tanımlanan süreç tanımlaması nesnesinin *UserData* özniteliğini kullanarak bu alanı doldurur. Bu alan, tetiklenecek kanalın adını belirtmek için kullanılabilir.

There is a full description of the trigger message structure in MQTM.

Tetikleme işe yaramadığında

Tetikleme izleme programı programı başlatamazsa ya da kuyruk yöneticisi tetikleme iletisini sunamazsa, program tetiklenmez. Örneğin, süreç nesnesindeki applid, programın artalanda başlatılacağını belirtmelidir; tersi durumda, tetikleme izleme programı programı başlatamaz.

Bir tetikleme iletisi yaratılırsa, ancak başlatma kuyruğuna (örneğin, kuyruk dolu olduğu için ya da tetikleyici iletisinin uzunluğu, başlatma kuyruğu için belirtilen ileti uzunluğu üst sınırından büyük olduğu için) konulamazsa, tetikleme iletisi, ölü harf (teslim edilmemiş ileti) kuyruğunda yerine konmaya neden olur.

Ölü-mektup kuyruğuna koyma işlemi başarıyla tamamlanamazsa, tetikleme iletisi atılır ve z/OS konsoluna ya da sistem işletmenine bir uyarı iletisi gönderilir ya da hata günlüğüne gönderilir.

Tetikleme iletisini ölüme ilişkin ileti kuyruğuna koymak, o kuyruk için bir tetikleyici iletisi oluşturabilir. Bu ikinci tetikleme iletisi, ölü-mektup kuyruğuna bir ileti eklediğinde atılır.

Program başarıyla tetiklenirse, ancak kuyruktan iletiyi almadan önce olağandışı sona ererse, başarısızlığın nedenini bulmak için izleme yardımcı programını kullanın (örneğin, program CICSaltında çalışıyorsa CICS AUXTRACE gibi).

MQI ve kümelerle çalışma

Aramalara ilişkin özel seçenekler ve kümeleme ile ilgili dönüş kodları vardır.

Aramalar ve kümelerle kullanılacak dönüş kodlarında kullanılabilir olan seçenekler hakkında daha fazla bilgi edinmek için aşağıdaki bağlantıları kullanın:

- “MQOPEN ve kümeler” sayfa 331
- “MQPUT, MQPUT1 ve kümeler” sayfa 332
- “MQINQ ve kümeler” sayfa 333
- “MQSET ve kümeler” sayfa 333
- “Dönüş kodları” sayfa 334

İlgili kavramlar

“Message Queue Interface-Genel Bakış” sayfa 187
Message Queue Interface (MQI) bileşenleri hakkında bilgi edinin.

“Kuyruk yöneticisine bağlanma ve kuyruk yöneticisinden bağlantı kesme” sayfa 198
WebSphere MQ programlama hizmetlerini kullanmak için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir kuyruk yöneticisinden bağlantı kurulabilmek ve kuyruk yöneticisinden nasıl bağlantı kurulacağını öğrenmek için bu bilgileri kullanın.

“Nesnelerin açılması ve kapatılması” sayfa 206
Bu bilgiler, WebSphere MQ nesnelerini açmak ve kapatmak için bir kavrayış sağlar.

“İletileri Kuyruğa Koyma” sayfa 215
İletilerin kuyruğa nasıl konacağını öğrenmek için bu bilgileri kullanın.

“Kuyruktan Ileti Alınması” sayfa 230
Kuyruktan ileti alma hakkında bilgi edinmek için bu bilgileri kullanın.

“Nesne özniteliklerinin sorulmasına ve ayarlanmasını geciktirme” sayfa 305
Öznitelikler, bir WebSphere MQ nesnesine ilişkin özellikleri tanımlayan özelliklerdir.

“İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308
Bu bilgilerde, bir iş biriminde ortaya çıkan kurtarılabilir alma ve koyma işlemlerinin nasıl kesinleştirileceği ve geri alınacağı açıklanır.

“Starting IBM WebSphere MQ applications using triggers” sayfa 313
Tetikleyiciler kullanılarak IBM WebSphere MQ uygulamalarının nasıl başlatılacağı ve tetikleyiciler hakkında bilgi edinin.

MQOPEN ve kümeler

Bir küme kuyruğunun açılması için bir iletinin konulduğu ya da okunduğu kuyruğun MQOPEN çağrısına bağlı olduğu kuyruk.

Hedef kuyruğun seçilmesi

If you do not provide a queue manager name in the object descriptor, MQOD, the queue manager selects the queue manager to send the message to. Nesne tanımlayıcısında bir kuyruk yöneticisi adı sağlıyorsanız, iletiler her zaman seçtiğiniz kuyruk yöneticisine gönderilir.

Kuyruk yöneticisi hedef kuyruk yöneticisini seçiyorsa, seçim, bağ tanımlama seçeneklerine (MQOO_BIND_*) ve yerel bir kuyruğun varsa, buna bağlıdır. Kuyruğun yerel bir eşgörünümü varsa, CLWLUSEQ özniteliği ANYolarak ayarlanmadıkça, uzak bir yönetim ortamı tercihinde her zaman açılmıştır. Tersi durumda, seçim, bağlama seçeneklerine bağlıdır. Gruptaki tüm iletilerin aynı hedefte işlendiğinden emin olmak için kümeler ile ileti grupları kullanıldığında MQOO_BIND_ON_OPEN ya da MQOO_BIND_ON_GROUP belirtilmelidir.

Kuyruk yöneticisi hedef kuyruk yöneticisini seçiyorsa, iş yükü yönetimi algoritmasını kullanarak çevrimsel sıralı bir şekilde yapar; İş yükü dengelemesikonusuna bakın.

MQOO_BIND_ON_OPEN
MQOPEN çağrısındaki MQOO_BIND_ON_OPEN seçeneği, hedef kuyruk yöneticisinin düzeltileceğini belirtir. Bir küme içinde aynı kuyruğun birden çok örneği varsa MQOO_BIND_ON_OPEN seçeneğini kullanın. MQOPEN çağrısından döndürülen nesne tanıtıcısını belirten kuyruğa gönderilen tüm iletiler, aynı kuyruk yöneticisine yönelir.

- İletilerin yakınlıkları varsa, MQOO_BIND_ON_OPEN seçeneğini kullanın. Örneğin, bir ileti grubunun tümü aynı kuyruk yöneticisi tarafından işlenecekse, kuyruğu açtığınızda MQOO_BIND_ON_OPEN değerini belirtin. IBM WebSphere MQ , kuyruk yöneticisini ve o kuyruğa konarak tüm iletiler tarafından alınacak rotayı düzelir.
- MQOO_BIND_ON_OPEN seçeneği belirtilirse, kuyruk seçilmek üzere yeni bir kuyruk örneği için yeniden açılmalıdır.

MQOO_BIND_NOT_FIXED
MQOPEN çağrısındaki MQOO_BIND_NOT_FIXED seçeneği, hedef kuyruk yöneticisinin düzeltilmediğini belirtir. Messages written to the queue specifying the object handle returned from the MQOPEN call are routed to a queue manager at MQPUT time on a message-by-message basis. Tüm iletilerinizi aynı hedefe yazılacak şekilde zorlamak istemiyorsanız MQOO_BIND_NOT_FIXED seçeneğini kullanın.

- MQOO_BIND_NOT_FIXED ve MQMF_SEGMENTATION_ALLOWED değerlerini aynı anda belirtmeyin. Bunu yapmazsanız, iletinizin bölümleri farklı kuyruk yöneticilerine teslim edilebilir ve küme boyunca dağılmış olabilir.

MQOO_BIND_ON_GROUP
Bir uygulamanın, aynı hedef yönetim ortamına bir ileti grubunun ayrılmasını istemesine izin verir. Bu seçenek yalnızca kuyruklar için geçerlidir ve yalnızca küme kuyruklarını etkiler. Bir küme kuyruğu olmayan bir kuyruk için belirtilirse, bu seçenek yoksayılır.

- MQPUT üzerinde MQPMO_LOGICAL_ORDER belirtildiğinde gruplar tek bir hedefe yöneltiliyor. MQOO_BIND_ON_GROUP belirtilirse, ancak bir ileti bir grubun parçası değilse, bunun yerine BIND_NOT_FIXY davranışı kullanılır.

MQOO_BIND_AS_Q_DEF
MQOO_BIND_ON_OPEN, MQOO_BIND_NOT_FIXED ya da MQOO_BIND_ON_GROUPdeğerini belirtmezseniz, varsayılan seçenek MQOO_BIND_AS_Q_DEFolur. MQOO_BIND_AS_Q_DEF kullanılması, kuyruk tanıtıcısı için kullanılan bağ tanımının DefBind kuyruk özniteliğinden alınacağını sağlar.

MQOPEN seçeneklerinin yakınlığı

The MQOPEN options MQOO_BROWSE, MQOO_INPUT_*, or MQOO_SET require a local instance of the cluster queue for MQOPEN to succeed.

The MQOPEN options MQOO_OUTPUT, MQOO_BIND_*, or MQOO_INQUIRE do not require a local instance of the cluster queue to succeed.

Çözülmüş kuyruk yöneticisi adı

Bir kuyruk yöneticisi adı MQOPEN saatinde çözüldüğünde, çözülen ad uygulamaya geri döndürülür. Uygulama, sonraki bir MQOPEN çağrısında bu adı kullanmaya çalışırsa, bu adı kullanmaya yetkili olmadığını ortaya koyabilir.

MQPUT, MQPUT1 ve kümeler

If MQOO_BIND_NOT_FIXED is specified on an MQOPEN the workload management routines chooses which destination MQPUT or MQPUT1 select.

MQOPEN çağrısında MQOO_BIND_NOT_FIXED belirtilirse, sonraki her MQPUT çağrısı, iletinin hangi kuyruk yöneticisinin gönderileceğini belirlemek için iş yükü yönetimi yordamını çağırır. Alınacak hedef ve rota, iletiyle ileti temelinde seçilir. İleti, ağ değişiminde koşullar ortaya konduktan sonra, hedef ve rota değişebilir. MQPUT1 çağrısı her zaman MQOO_BIND_NOT_FIXED yürürlükte olduğu gibi çalışır; yani, her zaman iş yükü yönetimi yordamını çağırır.

İş yükü yönetimi yordamı bir kuyruk yöneticisi seçtiğinde, yerel kuyruk yöneticisi koyma işlemini tamamlar. İleti farklı kuyruklara yerleştirilebilir:

1. Hedef, kuyruğun yerel yönetim ortağıysa, ileti yerel kuyruğun üzerine yerleştirilir.
2. Hedef, bir kümedeki kuyruk yöneticisiyse, ileti bir küme iletim kuyruğuna yerleştirilir.

3. Hedef, bir küme dışında bir kuyruk yöneticisiyse, ileti, hedef kuyruk yöneticisiyle aynı adı taşıyan bir iletim kuyruğuna yerleştirilir.

MQOPEN çağrısında MQOO_BIND_ON_OPEN belirtilirse, hedef ve rota önceden seçildiği için MQPUT çağrıları iş yükü yönetimi yordamını çağırmaz.

MQINQ ve kümeler

Hangi küme kuyruğunun sorgulansa, MQOO_INQUIREile birleştirdiğiniz seçeneklere bağlıdır.

Before you can inquire on a queue, open it using the MQOPEN call and specify MQOO_INQUIRE.

To inquire on a cluster queue, use the MQOPEN call and combine other options with MQOO_INQUIRE. Sorgulanabilen öznitelikler, küme kuyruğunun yerel yönetim ortamı olup olmadığına ve kuyruğun nasıl açıldığı ile ilgili olarak değişir:

- MQOO_BROWSE, MQOO_INPUT_*ya da MQOO_SET ile MQOO_INQUIRE ile birleştirilmesi, açılımın başarılı olması için küme kuyruğunun yerel bir yönetim ortamını gerektirir. Bu durumda, yerel kuyruklar için geçerli olan tüm öznitelikleri sorgulayabilirsiniz.
- MQOO_OUTPUT ile MQOO_INQUIREbirleştirilerek, önceki seçeneklerden hiçbiri belirtilirse, açılan yönetim ortamı aşağıdakilerden biri olur:
  - Yerel kuyruk yöneticisiyse, varsa, yönetim ortamı. Bu durumda, yerel kuyruklar için geçerli olan tüm öznitelikleri sorgulayabilirsiniz.
  - Yerel bir kuyruk yöneticisi yönetim ortamı yoksa, kümenin başka bir yerinde yönetim ortamı. Bu durumda yalnızca aşağıdaki öznitelikler sorgulanabilir. Bu durumda, QType özniteliği MQQT_CLUSTER değerini içerir.
    - DefBind
    - DefPersistence
    - DefPriority
    - InhibitPut
    - QDesc
    - QName
    - QType

To inquire on the DefBind attribute of a cluster queue, use the MQINQ call with the selector MQIA_DEF_BIND. Döndürülen değer MQBND_BIND_ON_OPEN ya da MQBND_BIND_NOT_FIXEDya da MQBND_BIND_ON_GROUPolur. Gruplarla gruplar kullanılırken MQBND_BIND_ON_OPEN ya da MQBND_BIND_ON_GROUP belirtilmeli.

To inquire on the KüME and CLUSNL attributes of the local instance of a queue, use the MQINQ call with the selector MQCA_CLUSTER_NAME or the selector MQCA_CLUSTER_NAMELIST.

Not: Bir küme kuyruğunu MQOPEN ' un bağlı olduğu kuyruğu düzeltmeden açarsanız, sonraki MQINQ çağrıları, küme kuyruğunun farklı eşgörünümlerini sorgulayabilir.

İlgili kavramlar

“Küme kuyruğu için MQOPEN seçeneği” sayfa 211

Kuyruk tanıtıcısı için kullanılan bağ tanımı, MQBND_BIND_ON_OPEN, MQBND_BIND_NOT_FIXEDya da MQBND_BIND_ON_GROUPdeğerini alabilen *DefBind* kuyruk özniteliğinden alınır.

MQSET ve kümeler

The MQOPEN option MQOO_SET option requires there to be a local instance of a cluster queue for MQSET to succeed.

You cannot use the MQSET call to set the attributes of a queue elsewhere in the cluster.

Küme öznitelikle tanımlanmış bir yerel diğer ad ya da uzak kuyruk açabilir ve MQSET çağrısını kullanabilirsiniz. Yerel diğer adın ya da uzak kuyruğun özniteliklerini ayarlayabilirsiniz. Hedef kuyruğun, farklı bir kuyruk yöneticisininde tanımlı bir küme kuyruğu olması önemli değildir.

Dönüş kodları

Kümelere özgü dönüş kodları

MQRC_CLUSTER_EXIT_ERROR (2266 X'8DA')

Bir küme kuyruğunu açmak için bir MQOPEN, MQPUTya da MQPUT1 çağrısı yayınlanır ya da bu iletiyi bir ileti üzerine yerleştirir. Bir kuyruk yöneticisinin ClusterWorkloadExit özniteliği tarafından tanımlanan küme iş yükü çıkışı beklenmedik bir şekilde başarısız olur ya da zaman içinde yanıt vermez.

A message is written to the system log on WebSphere MQ for z/OS giving more information about this error.

Bu kuyruk tanıtıcısı için sonraki MQOPEN, MQPUTve MQPUT1 çağrıları, ClusterWorkloadExit özniteliğinin boş olmasına rağmen işlenir.

MQRC_CLUSTER_EXIT_LOAD_ERROR (2267 X'8DB')

z/OSüzerinde, küme iş yükü çıkışı yüklenemez.

Sistem günlüğüne bir ileti yazılır ve ClusterWorkloadExit özniteliği boş olmasına rağmen işleme devam eder.

z/OSdışında bir altyapıda, bir kuyruk yöneticisine bağlanmak için MQCONN ya da MQCONNX çağrısı yayınlanır. Kuyruk yöneticisinin kuyruk yöneticisi ClusterWorkloadExit özniteliği tarafından tanımlanan küme iş yükü çıkışı yüklenemediğinden, çağrı başarısız olur.

MQRC_CLUSTER_PUT_INHIBITED (2268 X'8DC')

Bir küme kuyruğu için geçerli olarak MQOO_OUTPUT ve MQOO_BIND_ON_OPEN seçenekleri ile bir MQOPEN çağrısı yayınlanır. All the instances of the queue in the cluster are currently put-inhibited by having the InhibitPut attribute set to MQQA_PUT_INHIBITED. İleti alınabilmekte olan kuyruk örneği olmadığından, MQOPEN çağrısı başarısız olur.

Bu neden kodu, aşağıdakilerin her ikisi de doğru olduğunda üretilir:

- Kuyruğun yerel eşgörünümü yok. Yerel bir yönetim ortamı varsa, yerel yönetim ortamı engellenmiş olsa bile MQOPEN çağrısı başarılı olur.
- Kuyruk için küme iş yükü çıkışı yok ya da bir küme iş yükü çıkışı var, ancak bir kuyruk eşgörünümü seçmiyor. (Küme iş yükü çıkışı bir kuyruk eşgörünümü seçerse, o yönetim ortamı konulsa bile MQOPEN çağrısı başarılı olur.)

MQOPEN çağrısında MQOO_BIND_NOT_FIXED seçeneği belirtilirse, kümedeki tüm kuyruklar engellense bile arama başarılı olabilir. Ancak, sonraki bir MQPUT çağrısı, bu çağrı sırasında tüm kuyruklar hala engellenmiş olsa da başarısız olabilir.

MQRC_CLUSTER_RESOLUTION_ERROR (2189 X'88D')

1. Bir küme kuyruğunu açmak için bir MQOPEN, MQPUTya da MQPUT1 çağrısı yayınlanır ya da bu iletiyi bir ileti üzerine yerleştirir. Tam havuz kuyruk yöneticisinden bir yanıt gerekli olduğundan, ancak hiçbiri kullanılabilir durumda olmadığından, kuyruk tanımlaması doğru şekilde çözümlenemiyor.
2. An MQOPEN, MQPUT, MQPUT1 or MQSUB call is issued for a topic object specifying YAYINLAMA(ALL) or ALT KAPSAM(ALL). The cluster topic definition cannot be resolved correctly because a response is required from the full repository queue manager but none is available.

MQRC_CLUSTER_RESOURCE_ERROR (2269 X'8DD')

Bir küme kuyruğu için bir MQOPEN, MQPUTya da MQPUT1 çağrısı yayınlanır. Kümeleme için gereken bir kaynağı kullanma girişimi sırasında bir hata oluştu.

MQRC_NO_DESTINATIONS_AVAILABLE (2270 X'8DE')

Bir iletiyi küme kuyruğuna koymak için bir MQPUT ya da MQPUT1 çağrısı yayınlanır. Arama sırasında, artık kümede kuyruğun herhangi bir eşgörünümü yok. MQPUT başarısız olur ve ileti gönderilmez.

queue, kuyruğu açan MQOPEN çağrısında MQOO_BIND_NOT_FIXED belirtilmişse ya da iletiyi koymak için MQPUT1 kullanılırsa hata oluşabilir.

MQRC_STOPPED_BY_CLUSTER_EXIT (2188 X'88C')

Bir iletiyi küme kuyruğuna açmak ya da bir küme kuyruğuna yerleştirmek için MQOPEN, MQPUTya da MQPUT1 çağrısı yayınlanır. Küme iş yükü çıkışı çağrıyı reddeder.

İstemci uygulamaları yazılıyor

What you need to know to write client applications on WebSphere MQ.

Uygulamalar WebSphere MQ istemcisi ortamında oluşturulabilir ve çalıştırılabilir. Uygulama oluşturulmalı ve kullanılan WebSphere MQ MQI istemcisine bağlı olmalıdır. Uygulamaların oluşturulacağı ve bağlantılı olduğu yol, kullanılan platforma ve programlama diline göre değişiklik gösterir. İstemci uygulamalarının nasıl oluşturulacağı hakkında bilgi için bkz. “WebSphere MQ MQI istemcilerine ilişkin uygulamalar oluşturuluyor” sayfa 340.

Bir WebSphere MQ uygulamasını hem tam WebSphere MQ ortamında hem de bir WebSphere MQ MQI istemcisi ortamında, belirli koşullar karşılansa da, kodunuzu değiştirmeden çalıştırabilirsiniz. For more information on running your applications in the WebSphere MQ client environment, see “Uygulamaların IBM WebSphere MQ MQI istemci ortamında çalıştırılması” sayfa 342.

If you use the message queue interface (MQI) to write applications to run in a WebSphere MQ MQI client environment there are some additional controls to impose during an MQI call to ensure that the WebSphere MQ application processing is not disrupted. Bu denetimler hakkında daha fazla bilgi için bkz. “İstemci uygulamasındaki ileti kuyruğu arabiriminin (MQI) kullanılması” sayfa 336.

İstemci uygulamaları olarak diğer uygulama tiplerini hazırlama ve çalıştırma bilgileri için aşağıdaki konulara bakın:

- “ CICS ve Tuxedo uygulamalarının hazırlanması ve çalıştırılması” sayfa 354
- “ Microsoft Transaction Server uygulamalarının hazırlanması ve çalıştırılması” sayfa 38
- “ WebSphere MQ JMS uygulamalarının hazırlanması ve çalıştırılması” sayfa 356

İlgili kavramlar

“Uygulama geliştirme kavramları” sayfa 7
IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“IBM WebSphere MQ uygulamalarını tasarlama” sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

“Örnek WebSphere MQ programları” sayfa 92
Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

“Kuyruğa alma uygulaması yazılıyor” sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

“ WebSphere MQ' da Web hizmetlerini kullanma” sayfa 907

SOAP için IBM WebSphere MQ iletimi ya da HTTP için IBM WebSphere MQ köprüsü kullanılarak Web hizmetleri için IBM WebSphere MQ uygulamaları geliştirebilirsiniz.

"Yayınlama/abone olma uygulamaları yazılıyor" sayfa 266
Yayınlama/abone olma WebSphere MQ uygulamalarını yazmaya başlayın.

"IBM WebSphere MQ uygulaması oluşturulması" sayfa 409
Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

"Program hatalarının işlenmesi" sayfa 526
Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

İstemci uygulamasındaki ileti kuyruğu arabiriminin (MQI) kullanılması

Bu konular toplaması, WebSphere MQ uygulamanızın bir WebSphere MQ MQI istemci ortamında çalışması ve tam WebSphere MQ kuyruk yöneticisi ortamında çalıştırılması arasındaki farkları göz önünde bulundurur.

When you design an application, consider what controls you need to impose during an MQI call to ensure that the WebSphere MQ application processing is not disrupted.

İstemci uygulamasındaki bir iletinin boyutunu sınırlandırma

Kuyruk yöneticisinin ileti uzunluğu üst sınırı, ancak bir istemci uygulamasından aktarabileceğiniz ileti büyüklüğü üst sınırı kanal tanımlamasıyla sınırlıdır.

Kuyruk yöneticisinin ileti uzunluğu üst sınırı (MaxMsguzunluğu) özniteliği, o kuyruk yöneticisi tarafından işlenebilecek bir iletinin uzunluk üst sınışıdır.

z/OSdışındaki platformlarda, bir kuyruk yöneticisinin ileti uzunluğu üst sınırı özniteliğini artırabilirsiniz. Ayrıntılar ALTER QMGR' de verilmiştir.

MQINQ çağrısını kullanarak, kuyruk yöneticisi için MaxMsgLength değerini öğrenebilirsiniz.

MaxMsgLength özniteliği değiştirilirse, kuyruklar ve hatta iletiler, yeni değerden daha büyük bir uzunlukla birlikte, önceden kuyruklar ve hatta ileti olmadığı için denetim yapılmadan yapılır. Bu özniteliği değiştirdikten sonra, değişikliğin yürürlüğe girdiğinden emin olmak için uygulamaları ve kanalları yeniden başlatın. Daha sonra kuyruk yöneticisinin ya da kuyruğun MaxMsguzunluğunu aşan yeni iletiler üretilemez (kuyruk yöneticisi bölümlemesine izin verilmediği sürece).

Bir kanal tanımlamasındaki ileti uzunluğu üst sınırı, istemci bağlantısıyla iletebileceğiniz bir iletinin büyüklüğünü sınırlar. Bir WebSphere MQ uygulaması, MQPUT çağrısını ya da MQGET çağrısını bu iletiden daha büyük bir iletiyle kullanmayı denerse, uygulamaya bir hata kodu döndürülür. Kanal tanımlamasının ileti büyüklüğü üst sınırı değiştirgesi, istemci bağlantısı üzerinden MQCB kullanılarak tüketilebilecek ileti büyüklüğü üst sınırını etkilemez.

İstemci ya da sunucu kodlanmış karakter takımı tanıtıcısı (CCSID) seçilmesi

İstemciye ilişkin yerel CCSID değerini kullanın. Kuyruk yöneticisi gerekli dönüştürmeyi gerçekleştirir. CCSID ' yi geçersiz kılmak için MQCCSID ortam değişkenini kullanın. Uygulamanız birden çok PUT gerçekleştiriyorsa, ilk PUT işlemi tamamlandıktan sonra, MQMD ' nin CCSID ve kodlama alanlarının üzerine yazılabilir.

The data passed across the MQI from the application to the client stub must be in the local CCSID, encoded for the WebSphere MQ MQI client. Bağlı kuyruk yöneticisi verilerin dönüştürülemesini gerektiriyorsa, dönüştürme işlemi kuyruk yöneticisinden istemci destek kodu tarafından gerçekleştirilir.

The Java client in V7, however, can do the conversion if the queue manager is unable to do so. Bkz. "Java istemcisi bağlantıları için WebSphere MQ sınıfları" sayfa 641

İstemci kodu, istemcideki MQI 'yi geçen karakter verilerinin, o iş istasyonu için yapılandırılmış CCSID' de olduğunu varsayar. Bu CCSID desteklenmeyen bir CCSID ise ya da gereken CCSID değilse, bu komutlardan birini kullanarak MQCCSID ortam değişkeniyle geçersiz kılınabilir:

- Pencerelerüzerinde:

```
SET MQCCSID=850
```

- UNIXsistemlerinde:

```
export MQCCSID=850
```

Tanıtımda bu parametre belirlendiyse, tüm MQI verilerinin 850 kod sayfasında olduğu varsayılır.

Not: 850 kod sayfasıyla ilgili varsayım, iletteki uygulama verileri için geçerli değildir.

Uygulamanızın ileti tanımlayıcısı (MQMD) sonrasında WebSphere MQ üstbilgileri içeren birden çok PUT gerçekleştiriyorsa, ilk PUT tamamlandıktan sonra MQMD ' nin CCSID ve kodlama alanlarının üzerine yazıldığını unutmayın.

After the first PUT, these fields contain the value used by the connected queue manager to convert the WebSphere MQ headers. Uygulamanızın, değerleri gerektirdiği değerlere sıfırladığından emin olun.

İstemci aplikasyonunda MQINQ kullanılması

MQINQ kullanılarak sorgulanan bazı değerler istemci kodu tarafından değiştirilir.

CCSID
istemci CCSID değerine ayarlıdır, kuyruk yöneticisinden değil.

MaxMsgUzunluğu
kanal tanımlamasıyla sınırlandırılırsa azaltılır. Bu işlem aşağıdaki gibi olacaktır:

- Kuyruk tanımlamasında tanımlanan değer ya da
- Kanal tanımlamasında tanımlanan değer

Ek bilgi için MQINQbaşlıklı konuya bakın.

İstemci uygulamasında eşitleme noktası eşgüdümü kullanma

Temel istemcide çalışan bir uygulama MQCMIT ve MQBACK yayınlayabilir; ancak, Sync Point denetiminin kapsamı MQI kaynaklarıyla sınırlıdır. Genişletilmiş işlemsel istemciyle bir dış hareket yöneticisi kullanabilirsiniz.

WebSphere MQiçinde, kuyruk yöneticisinin rollerinden biri, bir uygulama içindeki eşitleme noktası denetimidir. Bir uygulama bir WebSphere MQ temel istemcisinde çalışıyorsa, bu uygulama MQCMIT ve MQBACK yayınlayabilir, ancak Sync Point denetiminin kapsamı MQI kaynaklarıyla sınırlıdır. Bir temel istemci ortamında WebSphere MQ komutu MQBEGIN geçerli değil.

Sunucuda tam kuyruk yöneticisi ortamında çalışan uygulamalar, bir hareket izleme programı aracılığıyla birden çok kaynağı (örneğin veritabanları) koordine edebilir. On the server you can use the Transaction Monitor supplied with WebSphere MQ products, or another transaction monitor such as CICS. Bir işlem izleyiciyi temel istemci uygulaması ile kullanamazsınız.

Bir WebSphere MQ genişletilmiş işlemsel istemcisiyle bir dış hareket yöneticisi kullanabilirsiniz. Bkz. Genişletilmiş işlemsel istemci nedir? ayrıntılı bilgi için.

İstemci uygulamasında okuma yazma işlevini kullanma

İstemci uygulaması, iletileri istemek zorunda kalmadan bir istemciye kalıcı olmayan iletilerin gönderilmesini sağlamak için bir istemcide önceden okuma olanağını kullanabilirsiniz.

İstemci bir sunucudan ileti gerektirdiğinde, sunucuya bir istek gönderir. Tükettiği iletilerin her biri için ayrı bir istek gönderir. Bir istemcinin, bu istek iletilerini göndermekten kaçınarak kalıcı olmayan iletileri tüketmesini artırmak için, ileride okuma kullanacak şekilde yapılandırılmış bir istemci olabilir. Önden okuma, bir uygulamanın istekte bulunmadan istemciye gönderilmesine izin verir.

İleriyi okumanın kullanılması, bir istemci uygulamasından kalıcı olmayan iletiler tüketirken performansı yükseltebilirler. Bu performans iyileştirmesi, hem MQI hem de JMS uygulamaları için kullanılabilir.

MQGET ya da zamanuyumsuz tüketim kullanan istemci uygulamaları, kalıcı olmayan iletiler tüketildiğinde performans iyileştirmelerinden yararlanırlar.

MQOPEN ile MQO_READ_AHEAD ' i çağırdığınızda, WebSphere MQ istemcisi yalnızca belirli koşullar karşılandığında önden okuma seçeneğini etkinleştirir. Bu koşullar şunları içerir:

- İstemci ve uzak kuyruk yöneticisi, WebSphere MQ Sürüm 7 ya da sonraki bir sürümde olmalıdır.
- İstemci uygulaması, iş parçacıklı WebSphere MQ MQI istemci kitaplıklarına göre derlenmeli ve bağlantılandırılmalıdır.
- İstemci kanalının TCP/IP protokolünü kullanması gerekir
- Kanal, hem istemci hem de sunucu kanalı tanımlamalarında sıfır dışında bir SharingConversations (SHARECNV) ayarına sahip olmalıdır.

Okuma seçeneği geçerli kılındığında, iletiler ileriye okuma arabelleği adı verilen istemcide bir bellek arabelleğiyle gönderilir. İstemcinin okuma yazma arabelleğiyle açık bir okuma arabelleği vardır; bu arabelleği, önceden okuma özelliği etkinleştirilmiş olarak açar. İleriye okuma arabelleğindeki iletiler kalıcı olarak saklanmaz. İstemci düzenli olarak sunucuyu, tükettiği veri miktarına ilişkin bilgilerle güncelleştirir.

Tüm seçenekler kullanılmak üzere desteklenmediği için, tüm istemci uygulaması tasarımları önden okuma özelliğini kullanmaya uygun değildir. Önceden okuma etkinleştirildiğinde, MQGET çağrıları arasında tutarlı olması için bazı seçeneklerin tutarlı olması gerekir. Bir istemci, seçim ölçütünü MQGET çağrıları arasında değiştirirse, ileriye okuma arabelleğindeki saklanan iletiler, istemci okuma yazma arabelleğindeki iplikçik olarak kalır. Daha fazla bilgi için bkz. “Kalıcı olmayan iletilerin performansını artırma” sayfa 247

Okuma öncesinde okuma yapılandırması, WebSphere MQ istemcisi yapılandırma dosyasının MessageBuffer kısmında belirtilen üç öznitelik, MaximumSize, PurgeTimeve UpdatePercentagetarafından denetlenir.

İstemci uygulamasına zamanuyumsuz konması kullanma

Zamanuyumsuz put kullanılması, bir uygulamanın kuyruk yöneticisinden yanıt beklemeden kuyruğa ileti yerleştirmesini sağlar. Bazı durumlarda ileti alışverişi başarımını artırmak için bunu kullanabilirsiniz.

Olağan durumda, bir uygulama bir iletiyi ya da iletileri bir kuyruğa koyduğunda, MQPUT ya da MQPUT1kullanılarak, uygulamanın kuyruk yöneticisinin MQI isteğini işlediğini doğrulamasını beklemek gerekir. Özellikle istemci bağ tanımlarını kullanan uygulamalar için ileti alışverişi başarımını artırabilir ve iletileri zamanuyumsuz olarak koymak yerine, büyük sayıda küçük iletileri bir kuyruğa yerleştiren uygulamalar için kullanabilirsiniz. Bir uygulama bir iletiyi zamanuyumsuz olarak yerleştirdiğinde, kuyruk yöneticisi her çağrımın başarısını ya da hatasını döndürmez, ancak bunun yerine düzenli aralıklarla hata denetimi yapabilirsiniz.

Bir iletiyi zamanuyumsuz olarak bir kuyruğa koymak için, MQPMO yapısının *Options* alanında MQPMO_ASYNC_RESPONSE seçeneğini kullanın.

Bir ileti zamanuyumsuz koyma için uygun değilse, kuyruğa zamanuyumlu bir şekilde konmaya başlanır.

MQPUT ya da MQPUT1için zamanuyumsuz koyma yanıtı istenirken, bir CompCode ve MQCC_OK ve MQRC_NONE iletisinin nedeni, iletinin bir kuyruğa başarıyla konulduğu anlamına gelmeyebilir. Her bir MQPUT ya da MQPUT1 çağrısının başarısı ya da başarısızlığı hemen döndürülmese de, zamanuyumsuz bir çağrı altında oluşan ilk hata, daha sonra MQSTAT çağrısı yoluyla saptlanabilir.

MQPMO_ASYNC_RESPONSE ile ilgili ayrıntılar için MQPMO seçenekleribaşlıklı konuya bakın.

Zamanuyumsuz Koyma örnek programı, kullanılabilecek bazı özellikleri gösterir. Programın özelliklerinin ve tasarımının ve nasıl çalıştırılacağı konusunda ayrıntılı bilgi için bkz. “Zamanuyumsuz Koy örnek programı” sayfa 109.

Bir istemci uygulamasında sohbetlerin paylaşılmasını kullanma

Sohbet paylaşımına izin verildiği bir ortamda, sohbetler bir MQI kanalı eşgörünümünü paylaşabilir.

Her ikisi de kanal tanımlama (MQCD) yapısının bir parçası olan SharingConversationsadlı iki alan tarafından yapılan paylaşımda paylaşım, kanal çıkış parametresinin (MQCXP) bir parçası olan iki

alan tarafından kontrol edilir. MQCD ' deki SharingConversations (SharingConversations) alanı, kanalla ilişkilendirilmiş bir kanal yönetim ortamını paylaşabilecek etkileşim sayısı üst sınırını belirleyen bir tamsayı değeridir. MQCXP ' deki SharingConversations (SharingConversations) alanı, kanal örneğinin paylaşılıp paylaşılmadığını belirten bir Boole değeridir.

Paylaşımın paylaşılmasına izin verilmediği bir ortamda, aynı MQCD ' leri belirten yeni istemci bağlantıları bir kanal yönetim ortamını paylaşmaz.

Aşağıdaki koşullar doğru olduğunda, kanal yönetim ortamını yeni bir istemci uygulaması bağlantısı paylaşacak:

- Kanal yönetim ortamının istemci bağlantısı ve sunucu bağlantısı uçları, sohbetleri paylaşmak üzere yapılandırıldı ve bu değerler kanal çıkışlar tarafından geçersiz kılınmadı.
- İstemci bağlantısı MQCD değeri (istemcide MQCONNX çağrısında ya da istemci kanal tanımlama çizelgesinden (CCDT) sağlanır), varolan kanal yönetim ortamı ilk kez kurulduğunda, istemcide sağlanan MQCD çağrısına ya da CCDT ' den sağlanan istemci bağlantısı MQCD değeriyle tam olarak eşleşir. Özgün MQCD ' nin daha sonra çıkışlar ya da kanal anlaşması yoluyla değiştirilebileceğini, ancak bu değişikliklerin yapılmadan önce istemci sistemine sağlanan değerle eşleşmenin yapıldığını unutmayın.
- Sunucu tarafındaki paylaşım etkileşimleri sınırı aşılmaz.

Yeni bir istemci uygulaması bağlantısı, bir kanal örneğini diğer etkileşimler ile paylaşarak çalıştırma ölçütleriyle eşleşirse, bu karar o konuşmada herhangi bir çıkışa çağrılmadan önce yapılır. Böyle bir etkileşimde bulunan çıkışlar, kanal örneğini diğer etkileşimler ile paylaşıp paylaşmadığını değiştiremez. Yeni kanal tanımlamasıyla eşleşen var olan kanal yönetim ortamı yoksa, yeni bir kanal yönetim ortamı bağlanır.

Kanal anlaşması, yalnızca kanal yönetim ortamındaki ilk etkileşim için gerçekleşir; kanal örneğine ilişkin anlaşmalı değerler o aşamada sabitlenir ve sonraki etkileşimler başlatıldığında değiştirilemez. TLS/SSL kimlik doğrulaması da yalnızca ilk etkileşim için gerçekleşir.

MQCD SharingConversations değeri, istemci bağlantısında ya da kanal yönetim ortamının sunucu bağlantısı sonundaki yuvadaki ilk etkileşim için tüm güvenlik, gönderme ya da alma çıkışları sırasında değiştirilirse, tüm bu çıkışlardan sonra sahip olduğu yeni değer, kanal örneğine ilişkin paylaşma etkileşimleri değerini belirlemek için kullanılır (en düşük değer öncelikli olarak uygulanır).

Sohbetlerin paylaşılması için kararlaştırılan değer sıfırsa, kanal örneği hiçbir zaman paylaşılamamaktadır. Bu alanı sıfır olarak ayarlayan diğer çıkış programları da kendi kanal yönetim ortamında benzer şekilde çalışır.

Sohbetlerin paylaşılması için kararlaştırılan değer sıfırdan büyükse, MQCXP SharingConversations , sonraki çağrılar için TRUE olarak ayarlandıysa, bu kanal örneğindeki diğer çıkış programlarının bu programla eşzamanlı olarak girilebileceğini gösterir.

Bir kanal çıkış programı yazdığınızda, bu programın etkileşim paylaşımını içerebilecek bir kanal yönetim ortamında çalıştırılıp çalıştırılmayacağını göz önünde bulundurun. Kanal yönetim ortamı etkileşimleri paylaşmayı gerektiriyorsa, bu etkiyi değiştirerek, MQCD alanlarını değiştirmenin diğer örneklerindeki etkiyi göz önünde bulundurun; tüm MQCD alanları, tüm paylaşım etkileşimleri arasında ortak değerlere sahiptir. Kanal yönetim ortamı kurulduktan sonra, çıkış programları MQCD alanlarını değiştirmeye çalışırsa sorunlarla karşılaşabilirler. Bunun nedeni, kanal yönetim ortamında çalışan çıkış programlarının diğer eşgörünümlerinin aynı alanları aynı anda değiştirme girişiminde bulunmaları olabilir. Çıkış programlarınızla bu durum ortaya çıkabiliyorsa, çıkış kodunuzda MQCD ' ye erişimi diziselleştirmeniz gerekir.

Sohbetleri paylaşmak üzere tanımlanmış bir kanalla çalışıyorsanız, ancak paylaşımın belirli bir kanal yönetim ortamında gerçekleşmesini istemiyorsanız, kanal yönetim ortamındaki ilk konuşmada kanal çıkışını başlattığınızda SharingConversations adlı MQCD değerini 1 ya da 0 olarak ayarlayın. SharingConversationsdeğerlerine ilişkin bir açıklama için bkz. SharingConversations .

Örnek

Paylaşımın paylaşılması etkin.

Bir çıkış programı belirten istemci bağlantısı kanal tanımlamasını kullanıyorsunuz.

Bu kanal ilk kez başlatıldığında, çıkış programı başlatıldığı sırada bazı MQCD parametrelerinden bazılarını değiştirmektedir. Bunlar kanalın üzerinde işlem görmektedir, yani kanalın çalıştırdığı tanım, şu anda sağlanan olandan farklı. MQCXP SharingConversations değiştirgesi TRUE olarak ayarlandı.

Uygulamanın bu kanalı kullanarak bağlantı kurmasını sağlayan etkileşim, aynı özgün kanal tanımlamasına sahip olduğu için, daha önce başlatılan kanal örneğinde çalışır. Uygulamanın ikinci kez bağlandığı kanal yönetim ortamı, ilk bağlandığı zaman ile aynı yönetim ortağıdır. Sonuç olarak, çıkış programı tarafından değiştirilmiş olan tanımlamaları kullanır. İkinci etkileşim için çıkış programı ilk kullanıma hazırlandığında, bu program MQCD alanlarını değiştirebilse de, kanal tarafından *hareket etmiyorlardı* . Bu aynı özellikler, kanal yönetim ortamını paylaşan sonraki tüm etkileşimler için de geçerlidir.

MQCONNX olanağının kullanılması

MQCNO yapısındaki bir kanal tanımlaması (MQCD) yapısı belirtmek için MQCONNX çağrısını kullanabilirsiniz.

Bu, çağıran istemci uygulamasının çalıştırma zamanında istemci bağlantı kanalının tanımını belirtmesini sağlar. Daha fazla bilgi için, MQCONNX çağrısında MQCNO yapısının kullanılmasıbaşlıklı konuya bakın. MQCONNX 'i kullandığınızda, sunucuda yayınlanan arama sunucu düzeyi ve dinleyici yapılandırmasına bağlıdır.

İstemciden MQCONNX 'i kullandığınızda, aşağıdaki seçenekler yoksayılır:

- MQCNO_STANDARD_BINDING
- MQCNO_FASTPATH_BINDING

Kullanabileceğiniz MQCD yapısı, kullanmakta olduğunuz MQCD sürüm numarasına bağlıdır. MQCD sürümlerine (MQCD_VERSION) ilişkin bilgi için MQCD Version(MQCD Sürümü) konusuna bakın. Örneğin, kanal çıkış programlarını sunucuya geçirmek için, MQCD yapısını kullanabilirsiniz. MQCD Sürüm 3 ya da sonraki bir sürümünü kullanıyorsanız, bu yapıyı kullanarak, bir çıkış dizisini sunucuya geçirebilirsiniz. Var olan bir çıkışı değiştirmek yerine, her işlem için bir çıkış ekleyerek, şifreleme ve sıkıştırma gibi aynı iletide birden çok işlem gerçekleştirmek için bu işlevi kullanabilirsiniz. MQCD yapısında bir dizi belirtmezseniz, tek çıkış alanları denetlenir. Kanal çıkışı programlarıyla ilgili daha fazla bilgi için bkz. “İleti alışverişi kanallarına ilişkin kanal çıkışı programları” sayfa 378.

MQCONNX üzerinde paylaşılan bağlantı tanıtıcıları

Paylaşılan bağlantı tanıtıcılarını kullanarak, aynı süreç içindeki farklı iş parçacıkları arasında çekme noktaları paylaşabilirsiniz.

Paylaşılan bir bağlantı tanıtıcısı belirttiğinizde, MQCONNX çağrısından döndürülen bağlantı tanıtıcısı, süreçteki herhangi bir iş parçacığında sonraki MQI çağrılarına aktarılabilir.

Not: Paylaşılan bağlantı tanıtıcılarını desteklemeyen bir sunucu kuyruk yöneticisine bağlanmak için, bir WebSphere MQ MQI istemcisinden paylaşılan bağlantı tanıtıcısını kullanabilirsiniz.

Daha fazla bilgi için “MQCONNX olanağının kullanılması” sayfa 340 başlıklı konuya bakın.

WebSphere MQ MQI istemcilerine ilişkin uygulamalar oluşturuluyor

Uygulamalar WebSphere MQ MQI istemcisi ortamında oluşturulabilir ve çalıştırılabilir. Uygulama oluşturulmalı ve kullanılan WebSphere MQ MQI istemcisine bağlı olmalıdır. Uygulamaların oluşturulacağı ve bağlantılı olduğu yol, kullanılan platforma ve programlama diline göre değişiklik gösterir.

Bir uygulama istemci ortamında çalıştırılacaksa, bu uygulamayı aşağıdaki tabloda gösterilen dillerde yazabilirsiniz:

Çizelge 47. İstemci ortamlarında desteklenen programlama dilleri

İstemci altyapısı	C	C++	COBOL	pTAL	RPG	Visual Basic
AIX	Evet	Evet	Evet			

Çizelge 47. İstemci ortamlarında desteklenen programlama dilleri (devamı var)						
İstemci altyapısı	**C**	**C++**	**COBOL**	**pTAL**	**RPG**	**Visual Basic**
HP Integrity NonStop Server	Evet		Evet	Evet		
HP-UX	Evet	Evet	Evet			
Linux	Evet	Evet	Evet			
Solaris	Evet	Evet	Evet			
Pencereler	Evet	Evet	Evet			Evet

Bu dillerdeki istemci uygulamaları oluşturmaya ya da oluşturmaya ilişkin yönergeler için ilgili konulara bakın.

C uygulamalarını WebSphere MQ MQI istemci koduyla bağlantılandırma

Having written your WebSphere MQ application that you want to run on the WebSphere MQ MQI client, you must link it to a queue manager.

Uygulamanızı bir kuyruk yöneticisinde iki şekilde bağlayabilirsiniz:

1. Doğrudan doğruya, kuyruk yöneticisinin uygulamanızın aynı iş istasyonunda olması gerekir.
2. Aynı ya da farklı bir iş istasyonunda kuyruk yöneticilerine erişmenizi sağlayan bir istemci kitaplığı dosyası

WebSphere MQ , her ortam için bir istemci kitaplığı dosyası sağlar:

AIX
İş parçacıklı uygulamalar içinlibmqic.a kitaplığı ya da iş parçacıklı uygulamalar için libmqic_r.a kitaplığı.

HP-UX
İş parçacıklı uygulamalar içinlibmqic.sl kitaplığı ya da iş parçacıklı uygulamalar için libmqic_r.sl kitaplığı.

Linux
İş parçacıklı uygulamalar içinlibmqic.so kitaplığı ya da iş parçacıklı uygulamalar için libmqic_r.so kitaplığı.

Solaris
libmqic.so.

Solaris için yalnızca WebSphere MQ MQI istemcisi kurulu olan bir iş istasyonundaki programları kullanmak istiyorsanız, bunları istemci kitaplığınla bağlantılamak üzere yeniden derlemeniz gerekir:

```
$ /opt/SUNWspro/bin/cc -o <prog> <prog> c -mt -lmqic \
-lsocket -lc -lnsl -ldl
```

Parametreler, gösterildiği gibi doğru sırayla girilmelidir.

Pencereler
MQIC32.LIB.

C++ uygulamalarını WebSphere MQ MQI istemci koduyla bağlantılandırma

İstemcide C + + içinde çalışacak uygulamalar yazabilirsiniz. Oluşturma yöntemleri, ortama göre değişir.

C++ uygulamalarınızı nasıl bağlayabilmeye ilişkin bilgi için bakınız: Building WebSphere MQ C++ programs.

C + + kullanmanın tüm yönleriyle ilgili tüm ayrıntılar için bkz. C++ kullanılması

COBOL uygulamalarını IBM WebSphere MQ MQI istemci koduyla bağlantılandırma

IBM WebSphere MQ MQI istemcisinde çalıştırmak istediğiniz bir COBOL uygulaması yazdığınızda, bu uygulamayı uygun bir kitaplıkla ilişkilendirmeniz gerekir.

IBM WebSphere MQ , her ortam için bir istemci kitaplığı dosyası sağlar:

AIX
Link your non-threaded COBOL application with the library libmqicb.a or threaded COBOL application with libmqicb_r.a.

HP-UX
Link your non-threaded COBOL application with the library libmqicb.sl or threaded COBOL application with libmqicb_r.sl.

Linux
Link your non-threaded COBOL application with the library libmqicb.so or threaded COBOL application with libmqicb_r.so.

Solaris
Link your non-threaded COBOL application with the library libmqicb.so or threaded COBOL application with libmqicb_r.so.

Windows
Uygulama kodunuzu, 32 bit COBOL için MQICCBB kitaplığınızla ilişkilendirin. Windows için IBM WebSphere MQ MQI istemcisi, 16 bitlik COBOL ' yi desteklemez.

Visual Basic uygulamalarını WebSphere MQ MQI istemci koduyla bağlantılandırma

Visual Basic uygulamalarını Pencerelerüzerinde WebSphere MQ MQI istemci kodu ile bağlantılayabilirsiniz.

Visual Basic uygulamanızı aşağıdaki içerme dosyalarıyla bağlantılayın:

CMQB.bas
MQI

CMQBB.bas
MQAI

CMQCFB.bas
PCF komutları

CMQXB.bas
Kanallar

İstemci dll 'in doğru otomatik seçilmesini sağlamak için, Visual Basic derleyicisinde istemci için `mqtype=2` ' yi ayarlayın:

MQIC32.dll
Windows 2000, Windows XP ve Windows 2003

Uygulamaların IBM WebSphere MQ MQI istemci ortamında çalıştırılması

Belirli koşulların karşılanabilmesi koşuluyla, bir IBM WebSphere MQ uygulamasını hem tam IBM WebSphere MQ ortamında hem de bir IBM WebSphere MQ MQI istemci ortamında kodunuzu değiştirmeden çalıştırabilirsiniz.

Bu koşullar şunlardır:

- Uygulamanın koşutzamanlı olarak birden çok kuyruk yöneticisine bağlanması gerekmez.
- Kuyruk yöneticisi adının başına bir `MQCONN` ya da `MQCONNX` çağrısında yıldız (*) öneki eklenmez.

- Uygulamanın, IBM WebSphere MQ MQI istemcisinde çalıştırılan uygulamalar nelerdir?içinde listelenen istisnalardan herhangi birini kullanması gerekmez.

Not: Bağlantı düzenleme sırasında kullandığınız kitaplıklar, uygulamanızın çalışması gereken ortamı belirler.

IBM WebSphere MQ MQI istemci ortamında çalışırken şunu unutmayın:

- IBM WebSphere MQ MQI istemci ortamında çalışan her uygulama, sunucularla ilgili kendi bağlantılarına sahiptir. Bir uygulama, bir MQCONN ya da MQCONNX çağrısı her yayınında bir sunucuyla bir bağlantı kurar.
- Bir uygulama, iletileri zamanuyumlu olarak gönderir ve gönderir. Bu, istemcinin istemcideki çıkışı ve bir tamamlanma kodunun döndürülmesi ve ağ üzerindeki neden kodunun döndürülmesi arasında bir bekleme anlamına gelir.
- Tüm veri dönüştürme işlemi sunucu tarafından yapılır, ancak makinenin konfigürasyonu tanımlanmış CCSID 'lerinin geçersiz kılınmasına ilişkin ek bilgi için MQCCSID ' ye de bakın.

Connecting IBM WebSphere MQ MQI client applications to queue managers

Bir IBM WebSphere MQ MQI istemci ortamında çalışan bir uygulama, çeşitli yollarla bir kuyruk yöneticisine bağlanabilir. Ortam değişkenlerini, MQCNO yapısını ya da istemci tanımlama çizelgesini kullanabilirsiniz.

Bir IBM WebSphere MQ istemci ortamında çalışan bir uygulama bir MQCONN ya da MQCONNX çağrısını yayınlarken, istemci bağlantıyı nasıl gerçekleştireceğini tanımlar. Bir IBM WebSphere MQ istemcisine bir uygulama tarafından MQCONNX çağrısı yayınlandığında, MQI istemcisi kitaplığı istemci kanalı bilgilerini aşağıdaki sırada arar:

1. MQCNO yapısının (sağlandıysa) *ClientConnOffset* ya da *ClientConnPtr* alanlarının içeriğini kullanma. Bu alanlar, istemci bağlantı kanalının tanımı olarak kullanılacak kanal tanımlama yapısını (MQCD) tanıtır. Bağlantı ayrıntıları, önceden bağlan bir çıkış kullanılarak geçersiz kılınabilir. Daha fazla bilgi için bkz “Bir havuzdaki bağlanma öncesi çıkışı kullanarak bağlantı tanımlarına gönderme yapılması” sayfa 404.
2. MQSERVER ortam değişkeni ayarlandıysa, tanımladığı kanal kullanılır.
3. Bir mqclient.ini dosyası tanımlıysa ve bir ServerConnectionParms içeriyorsa, tanımladığı kanal kullanılır. Daha fazla bilgi için bakınız: Configuring a client using a configuration file ve CHANITING stanza of the client configuration file.
4. MQCHLLIB ve MQCHLTAB ortam değişkenleri ayarlandıysa, işaret ettikleri istemci kanal tanımlama çizelgesi kullanılır.
5. Bir mqclient.ini dosyası tanımlıysa ve ChannelDefinitionDirectory ve ChannelDefinitionDosya öznitelikleri içeriyorsa, istemci kanal tanımlama çizelgesini bulmak için bu öznitelikler kullanılır. Daha fazla bilgi için bakınız: Configuring a client using a configuration file ve CHANITING stanza of the client configuration file.
6. Finally, if the environment variables are not set, the client searches for a client channel definition table with a path and name that are established from the DefaultPrefix in the mqs.ini file. İstemci tanımlama çizelgesi için arama başarısız olursa, istemci aşağıdaki yolları kullanır:
   - UNIX and Linux sistemleri: /var/mqm/AMQCLCHL.TAB
   - Windows: C:\Program Files\IBM\Websphere MQ\amqclchl.tab

Önceki listede açıklanan seçeneklerin ilki (MQCNO ' nun *ClientConnOffset* ya da *ClientConnPtr* alanlarını kullanarak), yalnızca MQCONNX çağrısı tarafından desteklenir. Uygulama MQCONNX yerine MQCONN kullanıyorsa, kanal bilgileri listede gösterilen sırada kalan beş yolla aranır. İstemci kanal bilgilerini bulamazsa, MQCONN ya da MQCONNX çağrısı başarısız olur.

Kanal adı (istemci bağlantısı için), başarılı olması için MQCONN ya da MQCONNX çağrısına ilişkin sunucuda tanımlı olan sunucu bağlantısı kanal adıyla eşleşmelidir.

If you receive an MQRC_Q_MGR_NOT_AVAILABLE return code from your application with an error message in the error log file of AMQ9517 -Dosya zarar görmüş, see Geçiş ve istemci kanal tanımlama çizelgeleri (CCDT).

İlgili kavramlar
İstemci kanal tanımlama çizelgesi
İlgili görevler
Sunucu ve istemci arasındaki bağlantıların yapılandırılması
İlgili başvurular
MQSERVER
MQCHLIB
MQCHLTAB

İstemci uygulamalarının ortam değişkenleri kullanılarak kuyruk yöneticilerine bağlanması

İstemci kanalı bilgileri, MQSERVER, MQCHLLIB ve MQCHLTAB ortam değişkenleri tarafından istemci ortamında çalışan bir uygulamaya sağlanabilir.

Bu değişkenlere ilişkin ayrıntılar için bakınız: MQSERVER, MQCHLLIB ve MQCHLTAB .

MQCNO yapısını kullanarak istemci uygulamalarının kuyruk yöneticilerine bağlanması

Kanala ilişkin tanımlamayı, MQCONNX çağrısının MQCNO yapısı kullanılarak sağlanan bir kanal tanımlama yapısında (MQCD) belirtebilirsiniz.

Daha fazla bilgi için MQCONNX çağrısında MQCNO yapısının kullanılması başlıklı konuya bakın.

İstemci uygulamaları kuyruk yöneticilerine istemci kanal tanımlama çizelgesi kullanılarak bağlanması

MQSC DEFINE CHANNEL komutunu kullanırsanız, sağladığınız ayrıntılar istemci kanal tanımlama çizelgesine (ccdt) yerleştirilir. MQCONN ya da MQCONNX çağrısının *QMgrName* değiştirgesinin içeriği, istemcinin hangi kuyruk yöneticisini bağlamaya bağlandığı saptar.

Bu dosyaya istemcinin erişeceği kanalı belirlemek için istemci tarafından erişilir. Birden fazla uygun kanal tanımlaması varsa, kanal seçimi, istemci kanal ağırlığı (CLNTWGHT) ve bağlantı benzerliği (BENZEŞIMI) kanalı özniteliklerinden etkilenir.

İstemci kanalı tanımlama çizelgesinin rolü

İstemci kanal tanımlama çizelgesi (CCDT), istemci bağlantı kanallarına ilişkin tanımlamaları içerir. Bu, özellikle istemci uygulamalarınızın çeşitli kuyruk yöneticilerine bağlanmanız gerekmesi durumunda yararlı olur.

İstemci kanalı tanımlama çizelgesi, bir kuyruk yöneticisi tanımladığınızda yaratılır.

Not: Aynı dosya, birden çok IBM WebSphere MQ istemcisi tarafından kullanılabilir. Bu dosyanın, MQCHLLIB ve MQCHLTAB IBM WebSphere MQ ortam değişkenlerini kullanarak farklı sürümlerine erişmenizi sağlar. Ortam değişkenleriyle ilgili bilgi için bkz. WebSphere MQ ortam değişkenlerinin kullanılması .

CCDT ' deki kuyruk yöneticisi grupları
İstemci kanalı tanımlama çizelgesinde (CCDT) bir bağlantı kümesi tanımlamak için *kuyruk yöneticisi grubu*olarak tanımlayabilirsiniz. Bir uygulamayı, kuyruk yöneticisi grubunun bir parçası olan bir kuyruk yöneticisine bağlayabilirsiniz. This can be done by prefixing the queue manager name on an MQCONN or MQCONNX call with an asterisk.

Aşağıdakiler nedeniyle birden çok sunucu makinesinden bağlantı tanımlamayı seçebilirsiniz:

- Kullanılabilirliği artırmak için çalışmakta olan kuyruk yöneticilerinden herhangi birine bir istemci bağlamak istiyorsunuz.
- Bir istemciyi, son kez başarıyla bağlandığınız kuyruk yöneticisine yeniden bağlamak istiyorsanız, ancak bağlantı başarısız olursa, farklı bir kuyruk yöneticisine bağlanın.
- İstemci programındaki MQCONN komutunu yeniden vererek, bağlantı başarısız olursa, istemci bağlantısını farklı bir kuyruk yöneticisinde yeniden deneyebilmeyi isteyebilirsiniz.

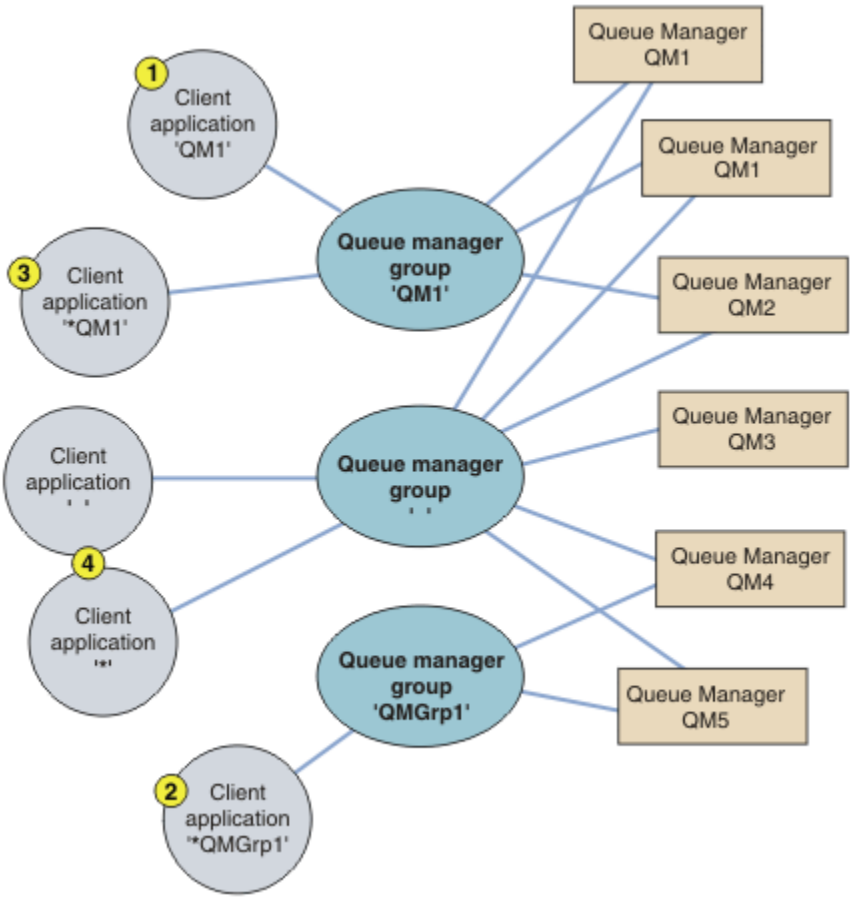
- Herhangi bir istemci kodu yazmadan, bağlantı başarısız olursa, istemci bağlantısını otomatik olarak başka bir kuyruk yöneticisine yeniden bağlamayı isteyebilirsiniz.
- Bir istemci kodunu yazmadan, yedek yönetim ortamı gerekiyorsa, çok eşgörünümli bir kuyruk yöneticisinin farklı bir eşgörünümesine otomatik olarak bağlanmak istiyorsunuz.
- İstemci bağlantılarınızı, bazı kuyruk yöneticileriyle ve bazı kuyruk yöneticilerine diğerlerinden daha fazla bağlanırken dengelemek istiyorsunuz.
- Çok sayıda istemci bağlantısının yeniden bağlanmasını birden çok kuyruk yöneticisi ve zaman içinde yeniden bağlamayı istiyorsanız, bu durumda, yüksek bağlantı hacmi bir hataya neden olur.
- Herhangi bir istemci uygulama kodunu değiştirmeden kuyruk yöneticilerinizi hareket ettirebilmeyi istiyorsunuz.
- Kuyruk yöneticisi adlarını bilmeye gerek olmayan istemci uygulama programları yazmak istiyorsunuz.

Her zaman farklı kuyruk yöneticilerine bağlanmak uygun değildir. An extended transactional client or a Java client in WebSphere Application Server, for example, might need to connect to a predictable queue manager instance. Otomatik istemci yeniden bağlanması, Java için WebSphere MQ sınıfları tarafından desteklenmez.

Kuyruk yöneticisi grubu, istemci kanal tanımlama çizelgesinde (CCDT) tanımlı bir bağlantı kümesidir. Küme, kendi kanal tanımlamalarında **QMNAME** özniteliğinin aynı değerine sahip üyeleri tarafından tanımlanır.

Şekil 70 sayfa 346 is a graphical representation of a client connection table, showing three queue manager groups, two named queue manager groups written in the CCDT as **QMNAME**(QM1) and **QMNAME**(QMGrp1), and one blank or default group written as **QMNAME**(' ').

1. Queue manager group QM1 has three client connection channels, connecting it to queue managers QM1 and QM2. QM1 , iki farklı sunucu üzerinde bulunan çok eşgörünümli bir kuyruk yöneticisi olabilir.
2. Varsayılan kuyruk yöneticisi grubunun, bunu tüm kuyruk yöneticilerine bağlayan altı istemci bağlantısı kanalı vardır.
3. QMGrp1 , iki kuyruk yöneticisine (QM4 ve QM5) istemci bağlantı kanallarına sahiptir.



Şekil 70. Kuyruk yöneticisi grupları

Bu istemci bağlantı çizelgesini kullanmanın dört örneği, Şekil 70 sayfa 346içindeki numaralandırılmış istemci uygulamalarının yardımlarıyla açıklanmaktadır.

1. İlk örnekte, istemci uygulaması bir kuyruk yöneticisi adını (QM1) MQCONN ya da MQCONNX MQI çağrısına **QmgrName** değiştirgesi olarak geçirir. WebSphere MQ istemci kodu, eşleşen kuyruk yöneticisi grubunu (QM1) seçer. The group contains three connection channels, and the WebSphere MQ MQI client tries to connect to QM1 using each of these channels in turn until it finds an WebSphere MQ listener for the connection attached to a running queue manager called QM1.

   Bağlantı denemelerinin sırası, istemci bağlantısı BENZEŞIMI özniteliğinin değerine ve istemci kanal ağırlıklandırmalarına bağlıdır. Bu kısıtlar içinde, bağlantı kurma yükünü dağıtmak için, her ikisi de olası üç bağlantı üzerinden ve zaman içinde bağlantı oluşturma sırası raslantılanmış olarak sıralanır.

   İstemci uygulaması tarafından yayınlanan MQCONN ya da MQCONNX çağrısı, çalışmakta olan bir QM1örneğine bağlantı kurulduğunda başarılı olur.
2. İkinci örnekte, istemci uygulaması başına yıldız imi olan bir kuyruk yöneticisi adını (*QMGrp1) MQCONN ya da MQCONNX MQI çağrısına **QmgrName** değiştirgesi olarak geçirir. The WebSphere MQ client selects the matching queue manager group, QMGrp1. Bu grup iki istemci bağlantı kanalı içerir ve WebSphere MQ MQI istemcisi, sırayla her bir kanalı kullanarak *any* kuyruk yöneticisine bağlanmayı dener. Bu örnekte, WebSphere MQ MQI istemcisinin başarılı bir bağlantı kurması gerekir; bağlandığı kuyruk yöneticisinin adı önemli değil.

   Bağlantı kurma girişimlerinin sırası daha önce de aynıdır. Tek fark, kuyruk yöneticisi adını yıldız işaretiyle önleyerek istemci, kuyruk yöneticisi adının ilgili olmadığını gösterir.

   İstemci uygulaması tarafından yayınlanan MQCONN ya da MQCONNX çağrısı, QMGrp1 kuyruk yöneticisi grubundaki kanallara bağlı herhangi bir kuyruk yöneticisinin çalışan bir eşgörünümünün çalışan bir eşgörünümüdür için bağlantı kurulduğunda başarılı olur.

3. The third example is essentially the same as the second because the **QmgrName** parameter is prefixed by an asterisk, *QM1. Bu örnek, bir kanal tanımlamasındaki QMNAME özniteliğini tek başına inceleyerek istemci kanalı bağlantısının hangi kuyruk yöneticisini denetleyeceğini belirleyemediğinizi gösterir. Kanal tanımlamasının **QMNAME** özniteliğinin QM1olması, QM1adlı bir kuyruk yöneticisine bağlantı yapılmasını zorunlu kılmamak için yeterli değildir. İstemci uygulamanız **QmgrName** parametresini bir yıldız işaretiyle önekler, sonra herhangi bir kuyruk yöneticisi olası bir bağlantı hedefidir.

   Bu durumda, istemci uygulaması tarafından yayınlanan MQCONN ya da MQCONNX çağrıları, çalışmakta olan bir QM1 ya da QM2yönetim ortamında bir bağlantı kurulduğunda başarılı olur.
4. Dördüncü örnek, varsayılan grubun kullanımını gösterir. Bu durumda, istemci uygulaması MQCONN ya da MQCONNX MQI çağrısında **QmgrName** değiştirgesi olarak bir yıldız imi ('*') ya da boş ''değerini geçirir. İstemci kanalı tanımlamasında kural olarak, boş bir **QMNAME** özniteliği, varsayılan kuyruk yöneticisi grubunu belirtir ve boşluk ya da yıldız işareti **QmgrName** değiştirgesi boş bir **QMNAME** özniteiyle eşleşir.

   Bu örnekte, varsayılan kuyruk yöneticisi grubunun tüm kuyruk yöneticilerine istemci kanalı bağlantıları vardır. Varsayılan kuyruk yöneticisi grubu seçilerek, uygulama gruptaki herhangi bir kuyruk yöneticisine bağlı olabilir.

   İstemci uygulaması tarafından yayınlanan MQCONN ya da MQCONNX çağrısı, herhangi bir kuyruk yöneticisinin çalışmakta olan bir eşgörünümüye bağlantı kurulduğunda başarılı olur.

   Not: Varsayılan bir kuyruk yöneticisinden varsayılan grup farklıdır; ancak, bir uygulama varsayılan kuyruk yöneticisi grubuna ya da varsayılan kuyruk yöneticisine bağlanmak için boş bir **QmgrName** değiştirgesi kullanıyor. Varsayılan kuyruk yöneticisi grubu kavramı yalnızca istemci uygulamalarıyla ilgilidir ve bir sunucu uygulamasına varsayılan kuyruk yöneticisi içerir.

İstemci bağlantı kanallarınızı yalnızca bir kuyruk yöneticisinden tanımlayın; bu kanallar, ikinci ya da üçüncü kuyruk yöneticisine bağlanan kanallar da içinde olmak üzere. *değil* ' u iki kuyruk yöneticisinden tanımlayın ve sonra iki istemci kanal tanımlama çizelgelerini birleştirmeyi deneyin. İstemci kanal tanımlama çizelgesine yalnızca bir istemci kanal tanımlama çizelgesi erişebilir.

Örnekler

Konunun başlangıcındaki kuyruk yöneticisi gruplarının kullanılmasına ilişkin nedenlerin listesine yeniden bakın. Bir kuyruk yöneticisi grubu bu yetenekleri nasıl sağlıyor?

Kuyruk yöneticisi kümenlerinden herhangi birine bağlanın.
Define a queue manager group with connections to all the queue managers in the set, and connect to the group using the **QmgrName** parameter prefixed by an asterisk.

Aynı kuyruk yöneticisine yeniden bağlanın, ancak son kez bağlanılan kuyruk yöneticisi kullanılmıyorsa, farklı bir kuyruk yöneticisine bağlanın.
Daha önce olduğu gibi bir kuyruk yöneticisi grubu tanımlayın, ancak her istemci kanalı tanımlamasındaki özniteliği **AFFINITY**(PREFERENT)olarak ayarlayın.

Bir bağlantı başarısız olursa, başka bir kuyruk yöneticiyle bağlantı kurmayı yeniden deneyin.
Bir kuyruk yöneticisi grubuna bağlanın ve bağlantı bozuk ya da kuyruk yöneticisi başarısız olursa, MQCONN ya da MQCONNX MQI çağrısını yeniden yayınlayın.

Bir bağlantı başarısız olursa otomatik olarak başka bir kuyruk yöneticisine yeniden bağlanın.
MQCONNX **MQCNO** seçeneğini MQCNO_RECONNECTkullanarak bir kuyruk yöneticisi grubuna bağlanın.

Çok eşgörünümli bir kuyruk yöneticisinin farklı bir eşgörünümesine otomatik olarak yeniden bağlanın.
Önceki örneğe benzer şekilde yapın. Bu durumda, kuyruk yöneticisi grubunu belirli bir çok eşgörünümli kuyruk yöneticisinin eşgörünümlerine bağlanmayı sınırlamak için sınırlamak istiyorsanız, bu grubu yalnızca çok eşgörünümli kuyruk yöneticisi yönetim ortamlarıyla bağlantılarla tanımlayın.

You can also ask the client application to issue its MQCONN or MQCONNX MQI call with no asterisk prefixed to the **QmgrName** parameter. Bu şekilde, istemci uygulaması yalnızca adlandırılmış kuyruk yöneticisine bağlanabilir. Son olarak, **MQCNO** seçeneğini MQCNO_RECONNECT_Q_MGRolarak ayarlayabilirsiniz. Bu seçenek, önceden bağlı olan aynı kuyruk yöneticisine yeniden bağlantı kabul eder. Bu değeri, normal bir kuyruk yöneticisinin aynı eşgörünümüyla yeniden bağlantıları sınırlandırmak için de kullanabilirsiniz.

Kuyruk yöneticilerindeki istemci bağlantılarınıdengelemek , bazı kuyruk yöneticilerine diğerlerinden daha fazla istemci bağlıyor.

Bir kuyruk yöneticisi grubu tanımlayın ve bağlantıları eşit olmayan bir şekilde dağıtmak için her istemci kanalı tanımlamasında **CLNTWGHT** özniteliğini ayarlayın.

Bir bağlantı ya da kuyruk yöneticisi hatasından sonra, istemci yeniden bağlantı yükünü eşit olmayan bir şekilde dağıt ve zaman içinde dağıt.

Önceki örneğe benzer şekilde yapın. WebSphere MQ MQI istemcisi, kuyruk yöneticilerindeki yeniden bağlantıları rasgele oluşturur ve zaman içinde yeniden bağlantıları dağıtır.

İstemci kodunu değiştirmeden kuyruk yöneticilerinizi taşıyın.

CCDT, istemci uygulamanızı kuyruk yöneticisinin yerinden yalıtmanızı sağlar.

İstemci bağlantı çizelgesini her istemciye dağıtarak ya da CCDT ' yi her bir istemci için paylaşılan bir kütük sistemine yerleştirmeyi seçmiş olabilir. Diğer bir seçenek olarak, MQCONNX MQI çağrısında desteklenen CCDT 'nin programsal sürümünü kullanın ve CCDT' yi istemci uygulamasına geçirmek için bir hizmeti çağırın.

Kuyruk yöneticisi adlarını tanımayan bir istemci uygulaması yazın.

Kuyruk yöneticisi grup adlarını kullanın ve kuruluşunuzda istemci uygulamalarınızla ilgili olan kuyruk yöneticisi grup adları için bir adlandırma kuralı oluşturun ve kuyruk yöneticilerinin adlandırılması yerine çözümlerinizin mimarisini yansıtır.

Kuyruk paylaşım gruplarıyla bağlantı kuruluyor

Uygulamanızı, kuyruk paylaşım grubunun bir parçası olan bir kuyruk yöneticisine bağlayabilirsiniz. Bu işlem, MQCONN ya da MQCONNX çağrısında kuyruk yöneticisi adı yerine, kuyruk paylaşım grubu adı kullanılarak yapılabilir.

Kuyruk paylaşım grupları en çok dört karakterden oluşan bir ada sahiptir. Adın ağınızda benzersiz olması ve kuyruk yöneticisi adlarından farklı olması gerekir.

İstemci kanalı tanımlaması, gruptaki kullanılabilir bir kuyruk yöneticisine bağlanmak için kuyruk paylaşım grubu soysal arabirimini kullanmalıdır. Daha fazla bilgi için İstemcinin bir kuyruk paylaşım grubuna bağlanmasıbaşlıklı konuya bakın. İletişimci, kuyruk yöneticisinin bağlandığı kuyruk yöneticisinin kuyruk paylaşım grubunun bir üyesi olduğundan emin olmak için bir onay imi yapılır.

Kanal ağırlıklandırma ve benzeşme örnekleri

Bu örnekler, sıfırsız `ClientChannelWeights` kullanıldığında istemci-bağlantı kanallarının nasıl seçildiğini gösterir.

`ClientChannelAğırlık` ve `ConnectionAffinity` kanalı öznitelikleri, bir bağlantı için birden çok uygun kanal kullanılabilir olduğunda istemci bağlantı kanallarının nasıl seçildiğini denetler. Bu kanallar, daha yüksek kullanılabilirlik, iş yükü dengelemesi ya da her ikisi için farklı kuyruk yöneticilerine bağlanmak üzere yapılandırılırmış. Bazı kuyruk yöneticilerinden biriyle bağlantının sonuçlanabileceği MQCONN çağrıları, şu konuda açıklandığı gibi, kuyruk yöneticisi adına bir yıldız işaretiyle önek olarak önek vermelidir:MQCONN çağrılarına ilişkin örnekler: Örnek 1. Kuyruk yöneticisi adı bir yıldız işareti (*)içeriyor.

Bir bağlantı için geçerli aday kanalları, QMNAME özniteliğinin MQCONN çağrısında belirlenen kuyruk yöneticisi adıyla eşleştiği durumlarda yer alıyor. Bir bağlantıya ilişkin tüm uygun kanalların `ClientChannelAğırlık` değeri sıfır (varsayılan) ise, örnekteki gibi alfabetik sırayla seçilir: MQCONN çağrılarına örnekler: Örnek 1. Kuyruk yöneticisi adı bir yıldız işareti (*)içeriyor.

Aşağıdaki örnekler, sıfır olmayan `ClientChannelWeights` kullanılmadığında ne olacağını göstermektedir. Bu özellik sözde rasgele kanal seçimini içerdiği için, örneklerin kesinlikle gerçekleştireceği gibi bir işlem dizisi gösterdiğine dikkat edin.

Örnek 1. ConnectionAffinity değeri TERCIH edilen olarak ayarlandığında kanal seçilmesi

This example illustrates how a WebSphere MQ MQI client selects a channel from a CCDT, where the ConnectionAffinity is set to PREFERRED.

Bu örnekte, bir kuyruk yöneticisi tarafından sağlanan bir İstemci Kanal Tanımlama Çizelgesi (CCDT) kullanan istemci makinelerinden oluşan bir dizi istemci makinelerinden biri. CCDT, aşağıdaki özniteliklere sahip istemci bağlantı kanallarını içerir (DEFINE CHANNEL komutunun sözdizimi kullanılarak gösterilir):

```
CHANNEL(A) QMNAME(DEV) CONNAME(devqm.it.company.example)
CHANNEL(B) QMNAME(CORE) CONNAME(core1.ops.company.example) CLNTWGHT(5) +
AFFINITY(PREFERRED)
CHANNEL(C) QMNAME(CORE) CONNAME(core2.ops.company.example) CLNTWGHT(3) +
AFFINITY(PREFERRED)
CHANNEL(D) QMNAME(CORE) CONNAME(core3.ops.company.example) CLNTWGHT(2) +
AFFINITY(PREFERRED)
```

Uygulama sorunları MQCONN (*CORE)

QMNAME özniteliği eşleşmediği için, Kanal A bu bağlantı için bir aday değil. B, C ve D kanalları aday olarak tanımlanır ve ağırlıklandırmalarına dayalı olarak bir tercih sırasına konur. Bu örnekte sıra C, B, D olabilir. İstemci, core2.ops.company.exampleadresindeki kuyruk yöneticisine bağlanmayı dener. MQCONN çağrısı kuyruk yöneticisi adına bir yıldız işareti eklediği için, o adresdeki kuyruk yöneticisinin adı denetlenmez.

AFFINITY(PREFERRED) ile, bu istemci makinesinde her bağlantı kesildiğinde, kanalları aynı ilk tercih sırasına göre yerleştirecek şekilde not almanız önemlidir. Bu, bağlantılar farklı süreçlerden ya da farklı zamanlarda olduğunda da geçerlidir.

Bu örnekte, core.2.ops.company.example konumundaki kuyruk yöneticisine ulaşılamıyor. İstemci core1.ops.company.example ile bağlantı kurmaya çalışır çünkü kanal B tercih sırasının yanında yer alıyor. Buna ek olarak, C kanalı en az tercih edilen düzeye indirgenir.

Aynı uygulama tarafından ikinci bir MQCONN (*CORE) çağrısı yayınlanıyor. Kanal C önceki bağlantı tarafından indirgenmiş, bu yüzden en çok tercih edilen kanal artık B ' dir. Bu bağlantı core1.ops.company.exampleolarak yapılır.

Aynı İstemci Kanal Tanımlaması Tablounu paylaşan ikinci bir makine, kanalları farklı bir başlangıç sırasına göre yerleştirmektedir. Örneğin, D, B, C. Normal koşullar altında, tüm kanallarda çalışan, bu makineden uygulamalar core3.ops.company.example ile bağlantılıysa, ilk makineden core2.ops.company.examplebağlantısı bulunur. Bu, her bir istemcinin kullanılabilir durumda olması durumunda aynı kuyruk yöneticisine bağlanmasını sağlarken, birden çok kuyruk yöneticisi arasında çok sayıda istemci için iş yükü dengelemesi yapılmasına olanak sağlar.

Örnek 2. ConnectionAffinity ' un NONE (Yok) olarak ayarlandığında kanal seçilmesi
Bu örnek, bir WebSphere MQ MQI istemcisinin CCDT 'den ConnectionAffinity ' un NONE (Yok) değerine ayarlandığı bir kanalı nasıl seçeceğini gösterir.

Bu örnekte, bir kuyruk yöneticisi tarafından sağlanan bir İstemci Kanal Tanımlama Çizelgesi (CCDT) kullanan sayıda istemci vardır. CCDT, aşağıdaki özniteliklere sahip istemci bağlantı kanallarını içerir (DEFINE CHANNEL komutunun sözdizimi kullanılarak gösterilir):

```
CHANNEL(A) QMNAME(DEV) CONNAME(devqm.it.company.example)
CHANNEL(B) QMNAME(CORE) CONNAME(core1.ops.company.example) CLNTWGHT(5) +
AFFINITY(NONE)
CHANNEL(C) QMNAME(CORE) CONNAME(core2.ops.company.example) CLNTWGHT(3) +
AFFINITY(NONE)
CHANNEL(D) QMNAME(CORE) CONNAME(core3.ops.company.example) CLNTWGHT(2) +
AFFINITY(NONE)
```

Uygulama, MQCONN (*CORE) sorunlarını yayınlar. Önceki örnekte olduğu gibi, QMNAME eşleşmediği için Kanal A dikkate alınmıyor. Kanal B, C ya da D, %50, %30 ya da %20 olasılıkları ile ağırlıklandırmalarına dayalı olarak seçilidir. Bu örnekte kanal B seçilebilir. Kalıcı bir tercih sırası oluşturulmadı.

İkinci bir MQCONN (*CORE) çağrısı yapıldı. Yine aynı olasılıklara sahip, uygulanabilir üç kanaldan biri seçiliyor. Bu örnekte, C kanalı seçilidir. Ancak, core2.ops.company.example yanıt vermez, bu nedenle kalan aday kanallar arasında başka bir seçenek de yapılır. Kanal B seçildi ve uygulama core1.ops.company.exampleile bağlantılıdır.

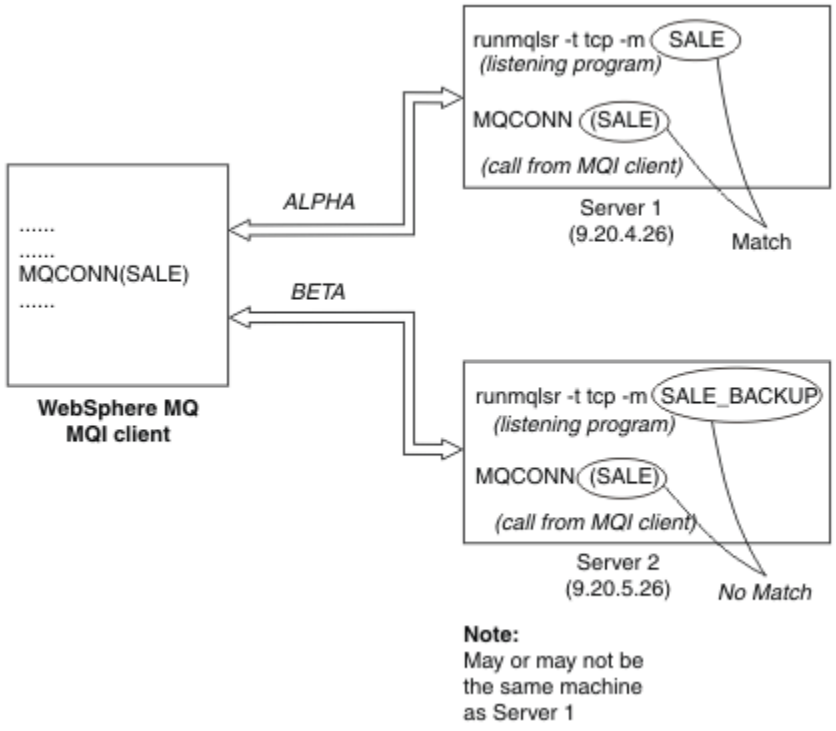
BENZEŞIMI (NONE) ile, MQCONN çağrılarının her biri diğerinden bağımsızdır. Bu nedenle, bu örnek uygulama üçüncü bir MQCONN (*CORE) yaptığında, B ya da D ' den birini seçmeden önce, bozuk kanal C ile bağlantı kurma girişiminde bulunmayı bir kez daha deneyebilir.

MQCONN çağrılarına örnekler

Belirli bir kuyruk yöneticisine ya da kuyruk yöneticilerinden birine bağlanmak için MQCONN kullanımına ilişkin örnekler.

Aşağıdaki örneklerin her birinde, ağ aynıdır; aynı WebSphere MQ MQI istemcisinden iki sunucuya tanımlanmış bir bağlantı vardır. (Bu örneklerde, MQCONN çağrısının yerine MQCONNX çağrısı kullanılabilir.)

Sunucu makinelerinde çalışan iki kuyruk yöneticisi vardır; biri SALE ve diğer adı SALE_BACKUP.



Şekil 71. MQCONN örneği

Bu örneklerdeki kanallara ilişkin tanımlamalar şunlardır:

SALE tanımlamaları:

```
DEFINE CHANNEL(ALPHA) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
DESCR('Server connection to WebSphere MQ MQI client')

DEFINE CHANNEL(ALPHA) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME(9.20.4.26) DESCR('WebSphere MQ MQI client connection to server 1') +
QMNAME(SALE)

DEFINE CHANNEL(BETA) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME(9.20.5.26) DESCR('WebSphere MQ MQI client connection to server 2') +
QMNAME(SALE)
```

SALE_BACKUP tanımlaması:

```
DEFINE CHANNEL(BETA) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
DESCR('Server connection to WebSphere MQ MQI client')
```

İstemci kanalı tanımlamaları aşağıdaki gibi özetlenebilir:

Ad	KLASöR	TRPTYPE	ADı	QMNAME
Alfa	NTNTCONN	TCP	9.20.4.26	SALE
Beta	NTNTCONN	TCP	9.20.5.26	SALE

MQCONN örneklerinin gösterisi
Örnekler, birden çok kuyruk yöneticisinin yedek sistem olarak kullanılmasını gösterir.

Sunucu 1 'e ilişkin iletişim bağlantısının geçici olarak bozulması olduğunu varsayın. Yedekleme sistemi olarak birden çok kuyruk yöneticisinin kullanılması gösterilir.

Her bir örnek, farklı bir MQCONN çağrısını kapsar ve aşağıdaki kuralları uygulayarak, sunulan belirli örnekte neler olduğuna ilişkin bir açıklama sunar:

1. İstemci kanal tanımlama çizelgesi (CCDT), MQCONN çağrısında belirtilen kuyruk yöneticisi adına (QMNAME alanı) karşılık gelen alfabetik kanal adı sırasından taranır.
2. Bir eşleşme bulunursa, kanal tanımlaması kullanılır.
3. Kanalı, bağlantı adı (CONNAME) ile tanıtılan makineye başlatma girişiminde bulunmanız gerekir. Bu başarılıysa, uygulama devam eder. Bu işlem aşağıdakileri gerektirir:
   - Sunucuda çalışmakta olan bir dinleyici.
   - İstemcinin bağlanmak istediği kuyruk yöneticisine (belirtilmişse) bağlı olan dinleyici.
4. Kanalı başlatma girişimi başarısız olursa ve istemci kanal tanımlama çizelgesinde birden çok giriş varsa (bu örnekte iki giriş varsa), dosya daha fazla eşleşme için aranır. Eşleşme bulunursa, işlem adım 1 'de devam eder.
5. Eşleşme bulunamazsa ya da istemci kanal tanımlama çizelgesinde başka giriş yoksa ve kanal başlatılamadıysa, uygulama bağlanamıyor demektir. MQCONN çağrısında, uygun bir neden kodu ve tamamlanma kodu döndürülür. Uygulama, döndürülen nedene ve tamamlanma kodlarına dayalı olarak işlem yapabilir.

Örnek 1. Kuyruk yöneticisi adı yıldız işareti (*) içeriyor
Bu örnekte, uygulama hangi kuyruk yöneticisinin bağlanacağı konusunda endişeli değildir. Uygulama, bir yıldız imi de dahil olmak üzere, kuyruk yöneticisi adı için bir MQCONN çağrısı yayınlar. Uygun bir kanal seçiliyor.

Uygulama sorunları:

```
MQCONN (*SALE)
```

Kurallardan sonra, bu örnekte olan budur:

1. The client channel definition table (CCDT) is scanned for the queue manager name SALE, matching with the application MQCONN call.
2. ALPHA ve BETA için kanal tanımları bulundu.
3. Bir kanalda CLNTWGHT değeri 0 ise, bu kanal seçilidir. If both have a CLNTWGHT value of 0, channel ALPHA is selected because it is first in alphabetical sequence. Her iki kanalda da sıfır olmayan CLNTWGHT değeri varsa, ağırlıklandırmasına dayalı olarak bir kanal rasgele seçilir.
4. Kanalı başlatma girişimi yapıldı.
5. If channel BETA was selected, the attempt to start it is successful.
6. ALPHA kanalı seçildiyse, iletişim bağlantısı bozuk olduğu için, bu kanal başlatılmaya çalışıldı. Aşağıdaki adımlar geçerli olur:
   a. The only other channel for the queue manager name SALE is BETA.
   b. Bu kanalı başlatma girişimi yapıldı; bu işlem başarılı oldu.
7. Bir dinleyicinin çalıştığını görmek için bir denetim, çalışmakta olan bir kişi olduğunu gösterir. Bu, SALE kuyruk yöneticisine bağlı değildir; ancak, MQI çağrısı parametresinin içinde bir yıldız (*) işareti bulunduğundan, herhangi bir denetim yapılmamıştır. Uygulama, SALE_BACKUP kuyruk yöneticisine bağlı ve işlemeye devam eder.

Örnek 2. Kuyruk yöneticisi adı belirtildi
Bu örnekte, uygulama belirli bir kuyruk yöneticisine bağlanmalıdır. Uygulama, o kuyruk yöneticisi adı için bir MQCONN çağrısı yayınlar. Uygun bir kanal seçiliyor.

The application requires a connection to a specific queue manager, named SALE, as seen in the MQI call:

```
MQCONN (SALE)
```

Kurallardan sonra, bu örnekte olan budur:

1. İstemci kanal tanımlama çizelgesi (CCDT), uygulama MQCONN çağrısıyla eşleşen SALEkuyruk yöneticisi adı için, alfabetik kanal adı sırasından taranır.
2. Eşleşmeyi içeren ilk kanal tanımlaması ALPHA.
3. Kanalı başlatma girişimi yapıldı; iletişim bağlantısı bozuk olduğu için bu *başarılı değil* .
4. The client channel definition table is again scanned for the queue manager name SALE and the channel name BETA is found.
5. Kanalı başlatma girişimi yapıldı; bu işlem başarılı oldu.
6. Bir dinleyicinin çalıştığını görmek için bir denetim, çalışmakta olan bir kişi olduğunu, ancak SALE kuyruk yöneticisine bağlı olmadığını gösterir.
7. İstemci kanal tanımlama çizelgesinde başka giriş yok. Uygulama devam edemiyor ve MQRC_Q_MGR_NOT_AVAM dönüş kodunu alıyor.

Örnek 3. Kuyruk yöneticisi adı boş ya da yıldız işareti (*)
Bu örnekte, uygulama hangi kuyruk yöneticisinin bağlanacağı konusunda endişeli değildir. Uygulama, boş bir kuyruk yöneticisi adı ya da yıldız işareti belirten bir MQCONN ile ilgili sorunları içerir. Uygun bir kanal seçiliyor.

Bu işlem, “Örnek 1. Kuyruk yöneticisi adı yıldız işareti (*) içeriyor” sayfa 351ile aynı şekilde ele alınır.

Not: Bu uygulama bir WebSphere MQ MQI istemcisi dışında bir ortamda çalışıyorsa ve ad boş bırakıldıysa, varsayılan kuyruk yöneticisine bağlanma girişiminde bulunulabilir. Bu, istemci ortamından çalıştırıldığı zaman *değildir* ; erişilen kuyruk yöneticisi, kanalın bağlandığı dinleyiciyle ilişkili olan bir yöneticidir.

Uygulama sorunları:

```
MQCONN ("")
```

ya da

```
MQCONN (*)
```

Kurallardan sonra, bu örnekte olan budur:

1. İstemci kanal tanımlama çizelgesi (CCDT), boş olan bir kuyruk yöneticisi adı için, MQCONN çağrısıyla eşleşen bir kuyruk yöneticisi adı için, alfabetik kanal adı sıralamasında taranır.
2. The entry for the channel name ALPHA has a queue manager name in the definition of SALE. This does *değil* match the MQCONN call parameter, which requires the queue manager name to be blank.
3. Sonraki giriş, BETAkanal adı içindir.
4. Tanımlamadaki `queue manager name` değeri SALE. Once again, this does *değil* match the MQCONN call parameter, which requires the queue manager name to be blank.
5. İstemci kanal tanımlama çizelgesinde başka giriş yok. Uygulama devam edemiyor ve MQRC_Q_MGR_NOT_AVAM dönüş kodunu alıyor.

İstemci ortamında tetikleme

WebSphere MQ MQI istemcilerinde çalışan WebSphere MQ uygulamaları tarafından gönderilen iletiler, diğer tüm iletilerle tam olarak aynı şekilde tetiklemeye katkıda bulunur ve bu iletiler hem sunucudaki, hem de istemcide programları tetiklemek için kullanılabilir.

Tetikleme, Tetikleme kanallarıiçinde ayrıntılı olarak açıklanmaktadır.

Tetikleme izleme programı ve başlatılacak uygulamanın aynı sistemde olması gerekir.

Tetiklenen kuyruğun varsayılan özellikleri, sunucu ortamındaki özelliklerle aynıdır. Özellikle, bir z/OS kuyruk yöneticisi tarafından yerel olan tetiklenen bir kuyruğa ileti koyan bir istemci uygulamasında MQPMO eşitleme noktası denetim seçeneği belirtilmediyse, iletiler bir iş birimi içinde yerleştirilir. Daha sonra tetikleme koşulu karşılanırsa, tetikleme iletisi aynı iş birimi içindeki başlatma kuyruğuna yerleştirilir ve iş birimi sona erinceye kadar tetikleyici izleme programı tarafından alınamaz. Tetiklenecek işlem, iş birimi sona erdirilinceye kadar başlatılmaz.

Süreç tanımlaması

Bu işlem, tetiklemeyi tetikleyen kuyrukla ilişkili olduğundan, sunucuda süreç tanımlamasını tanımlamalısınız.

Tetiklenecek olanları süreç nesnesi tanımlar. İstemci ve sunucu aynı altyapıda çalışmıyorsa, tetikleme izleyicisinin başlattığı tüm işlemler *ApplType*tanımlamalıdır; tersi durumda, sunucu varsayılan tanımlarını (yani, normalde sunucu makinesiyle ilişkili uygulama tipi) alır ve bir hataya neden olur.

Örneğin, tetikleme izleme programı bir Windows istemcisinde çalışıyorsa ve başka bir işletim sisteminde bir sunucuya istek göndermek istiyorsa, diğer bir işletim sistemi varsayılan tanımlamalarını kullanır ve işlem başarısız olur.

Tetikleyici İzleme Programı

z/OS dışı WebSphere MQ ürünleri tarafından sağlanan tetikleyici izleyicisi, UNIX, Linux ve Windows sistemleri için istemci ortamlarında çalıştırılır.

Tetikleme izleyicisini çalıştırmak için şu komutlardan birini çalıştırın:

- Linux UNIX Windows Windows, UNIX ve Linux altyapılarında:

```
runmqtmc [-m QMgrName] [-q InitQ]
```

Varsayılan başlatma kuyruğu SYSTEM.DEFAULT.INITIATION.QUEUE varsayılan kuyruk yöneticisininde KUYRUK. Başlatma kuyruğu, tetikleme izleyicinin tetikleme iletilerini göreceği yerdir. Daha sonra, uygun tetikleyici iletilerine ilişkin programları çağırır. Bu tetikleme izleme programı varsayılan uygulama tipini destekler ve istemci kitaplıklarını ilişkilendirmesi dışında, `runmqtrm` ile aynı olur.

Tetikleme izleme programı tarafından oluşturulan komut dizilimi aşağıdaki gibidir:

1. *ApplicId* , ilgili süreç tanımlamasından. *ApplicId* , çalıştırılacak programın adıdır; komut satırına girilir.
2. Başlangıç kuyruğundan elde edilen tırnak işaretleri içine alınmış `MQTMC2` yapısı. Sistem komutunun tek bir parametre olarak kabul ettiği sırayla, tam olarak sağlandığı şekilde, bu dizeye sahip bir komut dizisi başlatılır.
3. *EnvrData* , ilgili süreç tanımlamasından.

Tetikleme izleme programı, başlatma kuyruğunda başlatılmış olan uygulamanın tamamlanmasına kadar başka bir ileti olup olmadığını denetleyemez. Uygulamanın yapması gereken çok işlem varsa, tetikleme izleme programı gelen tetikleyici ileti sayısına yetişmeyebilir. Bu durumla başa çıkmak için iki yol vardır:

1. Çalışan daha fazla tetikleyici izleme programı var

   Daha fazla tetikleme izleme programının çalıştığını seçerseniz, herhangi bir zamanda çalışabilecek uygulama sayısı üst sınırını denetleyebilirsiniz.
2. Başlatılan uygulamaları arka planda çalıştır

   Uygulamaları arka planda çalıştırabilir, WebSphere MQ , çalışabilecek uygulama sayısı üzerinde herhangi bir kısıtlama oluşturmaz.

Başlatılan uygulamayı UNIX and Linux sistemlerinde arka planda çalıştırmak için, süreç tanımlamasının *EnvrData* sonuna bir & (ve işareti) koymanız gerekir.

CICS uygulamaları (z/OSdışı)

Bir MQCONN ya da MQCONNX çağrısı içerenz/OS dışı birCICS uygulama programı, CEDA olarak ASISTAN olarak tanımlanmalıdır. Bir CICS sunucusu uygulamasını istemci olarak yeniden bağlarsanız, eşitleme noktası desteğini kaybetmeyi riske atınız.

Bir MQCONN ya da MQCONNX çağrısı içerenz/OS dışı birCICS uygulama programı, CEDA olarak ASISTAN olarak tanımlanmalıdır. Yerleşik kodu mümkün olduğunca küçük yapmak için, MQCONN ya da MQCONNX çağrısını vermek için ayrı bir programa bağlantı yapabilirsiniz.

MQSERVER ortam değişkeni istemci bağlantısını tanımlamak için kullanıldıysa, bu değişkenin CICSENV.CMD dosyası.

WebSphere MQ applications can be run in a WebSphere MQ server environment or on a WebSphere MQ client without changing code. However, in a WebSphere MQ server environment, CICS can act as sync point coordinator, and you use EXEC CICS SYNCPOINT and EXEC CICS SYNCPOINT ROLLBACK rather than MQCMIT and MQBACK. Bir CICS uygulaması istemci olarak yeniden bağlantılandırılırsa, eşitleme noktası desteği kaybedilir. WebSphere MQ MQI istemcisi üzerinde çalışan uygulama içinMQCMIT ve MQBACK kullanılmalıdır.

CICS ve Tuxedo uygulamalarının hazırlanması ve çalıştırılması

CICS ve Tuxedo uygulamalarını istemci uygulamaları olarak çalıştırmak için, sunucu uygulamalarıyla kullandığınız kitaplıkları farklı kitaplıklarda kullanıyorsunuz. Uygulamanın çalıştırıldığı kullanıcı kimliği de farklıdır.

CICS ve Tuxedo uygulamalarını WebSphere MQ MQI istemci uygulamaları olarak çalışacak şekilde hazırlamak için, Genişletilmiş işlemsel istemcinin yapılandırılmasıbaşlıklı konu yönergelerinde yer alan yönergeleri izleyin.

Note, however, that the information that deals specifically with preparing CICS and Tuxedo applications, including the sample programs supplied with WebSphere MQ, assumes that you are preparing applications to run on a WebSphere MQ server system. Sonuç olarak, bilgiler yalnızca, bir sunucu sisteminde kullanılmak üzere hazırlanmış olan WebSphere MQ kitaplıklarına başvurur. İstemci uygulamalarınızı hazırlarken aşağıdaki şeyleri yapmanız gerekir:

- Uygulamanızın kullandığı dil bağlamaları için uygun istemci sistemi kitaplığını kullanın. Örneğin, C ' de AIX, HP-UXya da Solaris 'te yazılan uygulamalar için, libmqm yerine libmqic kitaplığını kullanın. Windows sistemlerinde, mqm.libyerine mqic.lib kitaplığını kullanın.
- Çizelge 48 sayfa 354içinde gösterilen sunucu sistemi kitaplıkları yerine, AIX, HP-UX, Solaris ve Çizelge 49 sayfa 354için, Windows sistemleri için eşdeğer istemci sistemi kitaplıklarını kullanın. Bu çizelgelerde listelenmeyen bir sunucu sistemi kitaplığı varsa, istemci sisteminde aynı kitaplığı kullanın.

Çizelge 48. AIX, HP-UXve Solaris üzerinde istemci sistemi kitaplıkları

Bir WebSphere MQ sunucu sistemine ilişkin kitaplık	**Bir WebSphere MQ istemcisi sisteminde kullanılacak eşdeğer kitaplık**
libmqmxa	libmqcxa

Çizelge 49. Pencereler sistemlerindeki istemci sistemi kitaplıkları

Bir WebSphere MQ sunucu sistemine ilişkin kitaplık	**Bir WebSphere MQ istemcisi sisteminde kullanılacak eşdeğer kitaplık**
mqmxa.lib	mqcxa.lib
mqmtux.lib	mqcxa.lib
mqmenc.lib	mqcxa.lib
mqmcics4.lib	mqccics4.lib

İstemci uygulaması tarafından kullanılan kullanıcı kimliği

CICS altında bir WebSphere MQ sunucusu uygulaması çalıştırdığınızda, normalde CICS kullanıcısından işlemin kullanıcı kimliğine geçer. Ancak, CICS altında bir WebSphere MQ MQI istemcisi uygulaması çalıştırdığınızda, CICS ayrıcalıklı yetkisine sahip olur.

CICS ve Tuxedo örnek programları

AIX, HP-UX, Solaris ve Windows sistemlerinde kullanılmak üzere CICS ve Tuxedo örnek programları.

Çizelge 50 sayfa 355 , AIX, HP-UXve Solaris istemci sistemlerinde kullanılmak üzere sağlanan CICS ve Tuxedo örnek programlarını listeler. Çizelge 51 sayfa 355 , Windows istemci sistemlerine ilişkin eşdeğer bilgileri listeler. Çizelgeler ayrıca, programları hazırlamak ve çalıştırmak için kullanılan dosyaları da listeler. Örnek programların bir açıklaması için bkz. "CICS hareket örneği" sayfa 112 ve "SMOKIN örnekleri" sayfa 148.

Çizelge 50. AIX, HP-UXve Solaris istemci sistemleri için örnek programlar

Tanım	Kaynak	Yürütülebilir modül
CICS programı	amqscic0.ccs	amqscicc
CICS programına ilişkin üstbilgi dosyası	amqscih0.h	-
İletileri koymak için smokin istemci programı	amqstxpx.c	-
İletileri almak için smokin istemci programı	amqstxgx.c	-
İki istemci programı için smokin sunucu programı	amqstxsx.c	-
Tuxedo programları için UBBCONFIG dosyası	ubbstxcx.cfg	-
Tuxedo programlarına ilişkin alan tablosu dosyası	amqstxvx.flds	-
Tuxedo programlarına ilişkin açıklama dosyasını görüntüle	amqstxvx.v	-

Çizelge 51. Pencereler istemci sistemleri için örnek programlar

Tanım	Kaynak	Yürütülebilir modül
CICS hareketi	amqscic0.ccs	amqscicc
CICS hareketine ilişkin üstbilgi dosyası	amqscih0.h	-
İletileri koymak için smokin istemci programı	amqstxpx.c	-
İletileri almak için smokin istemci programı	amqstxgx.c	-
İki istemci programı için smokin sunucu programı	amqstxsx.c	-
Tuxedo programları için UBBCONFIG dosyası	ubbstxcx.cfg	-
Tuxedo programlarına ilişkin alan tablosu dosyası	amqstxvx.fld	-
Tuxedo programlarına ilişkin açıklama dosyasını görüntüle	amqstxvx.v	-
Tuxedo programları için makefile	amqstxmc.mak	-
Tuxedo programlarına ilişkin ENVFILE dosyası	amqstxen.env	-

CICS ve Tuxedo uygulamaları için değişiklik olarak AMQ5203hata iletisi

Genişletilmiş bir işlemsel istemciyi kullanan CICS ya da Tuxedo uygulamalarını çalıştırdığınızda, standart tanılama iletilerini görebilirsiniz. Bunlardan biri, genişletilmiş bir işlemsel istemciyle kullanılmak üzere değiştirildi.

WebSphere MQ hata günlüğü dosyalarında görebileceğiniz iletiler Tanılama iletilerinde: AMQ4000-9999belgesinde belgelenir. Message AMQ5203 has been modified for use with an extended transactional client. Değiştirilen iletinin metni şöyledir:

AMQ5203: XA arabirimi çağrılırken bir hata oluştu.

Açıklama
Hata numarası & 2; 1 değeri, verilen işaret değerinin & 1 değerinin geçersiz olduğunu, 2 aynı işlemde iş parçacıklı ve iş parçacıklı olmayan kitaplıkları kullanma girişiminde bulunulduğunu, 3 ise, belirtilen kuyruk yöneticisi adı '& 3' ile ilgili bir hata olduğunu gösterir; 4, & 1 kaynak yöneticisi tanıtıcısının geçersiz olduğunu, 5 ise ikinci bir kuyruk yöneticisini kullanmak için girişimde bulunulduğunu gösterir. '& 3 'başka bir kuyruk yöneticisi önceden bağlandığında, 6, uygulama bir kuyruk yöneticisine bağlı olmadığı halde Transaction Manager 'ın çağrıldığını, 7, başka bir çağrı devam ederken XA çağrısının yapıldığını, 8 ise xa_open çağrısında' & 4 'xa_info dizgisinin' & 5 'parametre adı için geçersiz bir parametre değeri içerdiğini, 9 ise xa_open çağrısındaki' & 4 'xa_info dizgisinin gerekli bir parametre,' & 5 ' parametre adı eksik olduğunu gösterir.

Kullanıcı yanıtı
Hatayı düzeltin ve işlemi yeniden deneyin.

Microsoft Transaction Server uygulamalarının hazırlanması ve çalıştırılması

Bir MTS uygulamasını WebSphere MQ MQI istemci uygulaması olarak çalıştırmak üzere hazırlamak için, bu yönergeleri ortamınız için uygun olan yönergeleri izleyin.

For general information about how to develop Microsoft Transaction Server (MTS) applications that access WebSphere MQ resources, see the section on MTS in the WebSphere MQ Help Center.

Bir MTS uygulamasını WebSphere MQ MQI istemci uygulaması olarak çalıştırmak üzere hazırlamak için, uygulamanın her bileşeni için aşağıdakilerden birini yapın:

- Bileşen, MQI için C dili bağ tanımlarını kullanıyorsa, “Windows 'ta C programlarının hazırlanması” sayfa 440 içindeki yönergeleri izleyin, ancak bileşeni mqic.libyerine mqicxa.lib kitaplığıyla bağlantılayın.
- Bileşen, WebSphere MQ C++ sınıflarını kullanıyorsa, “C++ programlarını Windowsüzerinde oluşturma” sayfa 627 içindeki yönergeleri izleyin, ancak bileşeni imqc23vn.libyerine imqx23vn.lib kitaplığıyla bağlantılayın.
- Bileşen, MQI için Visual Basic dili bağ tanımlarını kullanıyorsa, “Visual Basic Programlarinin Windows 'ta Hazırlanması” sayfa 443 içindeki yönergeleri izleyin, ancak Visual Basic projesini tanımlarken **Koşullu Derleme Bağımsız Değişkenleri** alanında `MqType=3` yazın.
- If the component uses the WebSphere MQ Automation Classes for ActiveX (MQAX), define an environment variable, GMQ_MQ_LIB, with the value `mqic32xa.dll` .

  Ortam değişkenini uygulamanızın içinden tanımlayabilir ya da kapsamı sistem çapında olacak şekilde tanımlayabilirsiniz. Ancak, sistemi geniş olarak tanımlamak, var olan MQAX uygulamasının, uygulama içinden ortam değişkenini tanımlamamasına, yanlış şekilde davranmasına neden olabilir.

WebSphere MQ JMS uygulamalarının hazırlanması ve çalıştırılması

WebSphere MQ JMS uygulamalarını istemci kipinde çalıştırmak için WebSphere Application Server 'ı hareket yöneticiniz olarak çalıştırabilirsiniz. Bazı uyarı iletileri görebilirsiniz.

WebSphere MQ JMS uygulamalarını istemci kipinde hazırlamak ve çalıştırmak için, işlem yöneticiniz olarak WebSphere Application Server ile “JMS için WebSphere MQ sınıflarının kullanılması” sayfa 687başlıklı konu ile ilgili yönergeleri izleyin.

Bir WebSphere MQ JMS istemcisi uygulaması çalıştırdığınızda, aşağıdaki uyarı iletilerini görebilirsiniz:

MQJE080
Yetersiz lisans birimi-setmqcap komutunu çalıştırın

MQJE081
Lisans birimi bilgilerini içeren dosya yanlış biçimde, setmqcap komutunu çalıştırın.

MQJE082
Lisans birimi bilgilerini içeren dosya bulunamadı-setmqcap dosyasını çalıştırın

Kullanıcı çıkışları, API çıkışlar ve WebSphere MQ kurulabilir hizmetleri

Kuyruk yöneticisi olanaklarını kullanıcı çıkışlarını, API çıkışlarını ya da kurulabilir hizmetleri kullanarak genişletebilirsiniz. Bu konu, bu programların kullanılmasıyla ve geliştirilmesiyle ilgili bilgilere bağlantılar içerir.

Kuyruk yöneticisi olanaklarını genişletmek için kullanıcı çıkışlarını, API çıkışlarını ve kurulabilir hizmetleri nasıl kullanabileceğiyle ilgili bir giriş için bkz. Extending queue Manager olanakları.

Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesine ilişkin bilgi edinmek için “Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357başlıklı konuya bakın.

İlgili kavramlar
MQI kanallarına ilişkin kanal-çıkış programları
İlgili başvurular
API çıkış başvurusu
Kurulabilir hizmetler arabirimi başvuru bilgileri

Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi

UNIX, Linux ve Windows üzerindeki herhangi bir IBM WebSphere MQ kitaplığına bağlanmadan çıkışlar yazabilir ve derleyebilirsiniz.

Bu görev hakkında

Linux UNIX Windows Bu konu yalnızca Windows, UNIX and Linux sistemleri için geçerlidir. Diğer platformlara ilişkin çıkışlar ve kurulabilir hizmetlerle ilgili ayrıntılar için ilgili platforma özgü konulara bakın.

IBM WebSphere MQ varsayılan olmayan bir konuma kurulduysa, tüm IBM WebSphere MQ kitaplıklarına bağlanmadan çıkışlarınızı yazmanız ve derlemeniz gerekir.

Windows, UNIX and Linux sistemlerinde, bu IBM WebSphere MQ kitaplıklarından herhangi birini bağlantılandırmadan çıkışlar yazabilir ve derlenebilirsiniz:

- mqmzf
- mqm
- mqmvx
- mqmvxd
- mqic
- mqutl

Existing exits that are linked to these libraries continue to work, providing that on UNIX and Linux systems IBM WebSphere MQ is installed in the default location.

Yordam

1. cmqec.h üstbilgi dosyasını ekleyin.
   Bu üstbilgi dosyası otomatik olarak cmqc.h, cmqxc.h ve cmqzc.h üstbilgi kütüklerini içerir.
2. Çıkışı, MQI ve DCI çağrılarının MQIEP yapısıyla yapılıp yapılmaması için yazın. MQIEP yapısıyla ilgili ek bilgi için MQIEP yapısıbaşlıklı konuya bakın.
   - Kurulabilir hizmetler

  - MQZEP çağrısını göstermek için **Hconfig** parametresini kullanın.
  - **Hconfig** parametresini kullanmadan önce, **Hconfig** ' un ilk 4 baytının MQIEP yapısının **StrucId** ile eşleşmesini denetlemeniz gerekir.
  - Kurulabilir hizmet bileşenleri yazılmasıyla ilgili ek bilgi için MQIEPbaşlıklı konuya bakın.
- API çıkışları
  - MQXEP çağrısını göstermek için **Hconfig** parametresini kullanın.
  - **Hconfig** parametresini kullanmadan önce, **Hconfig** ' un ilk 4 baytının MQIEP yapısının **StrucId** ile eşleşmesini denetlemeniz gerekir.
  - API çıkışları yazma hakkında daha fazla bilgi için, bkz. “API çıkışları yazılıyor” sayfa 371.
- Kanal çıkışları
  - MQI ve DCI çağrılarını işaret etmek için MQCXP yapısının **pEntryPoints** parametresini kullanın.
  - **pEntryPoints**kullanılmadan önce MQCXP sürüm numarasının sürüm 8 ya da daha yüksek bir sürüm olduğunu doğrulayın.
  - Kanal çıkışlarını yazma hakkında daha fazla bilgi için bkz. “Kanal-çıkış programları yazılıyor” sayfa 381.
- Veri dönüştürme çıkışları
  - MQI ve DCI çağrılarını işaret etmek için MQDXP yapısının **pEntryPoints** parametresini kullanın.
  - **pEntryPoints**kullanılmadan önce MQDXP sürüm numarasının sürüm 2 ya da daha yüksek bir sürüm olduğunu doğrulayın.
  - You can use the **crtmqcvx** command and the amqsvfc0.c source file to create data conversion code that uses the **pEntryPoints** parameter. Bkz. “ WebSphere MQ for Windowsiçin veri dönüştürme çıkışı yazılıyor” sayfa 402 ve “UNIX and Linux sistemlerinde WebSphere MQ için veri dönüştürme çıkışı yazılıyor” sayfa 398.
  - **crtmqcvx** komutu kullanılarak oluşturulan veri dönüştürme çıkışların varsa, güncellenen komutu kullanarak çıkışı yeniden oluşturmalısınız.
  - Veri dönüştürme çıkışları yazma hakkında daha fazla bilgi için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397.
- Bağlantı öncesi çıkışlar
  - MQI ve DCI çağrılarını işaret etmek için MQNXP yapısının **pEntryPoints** parametresini kullanın.
  - **pEntryPoints**komutunu kullanmadan önce, MQNXP sürüm numarasının sürüm 2 ya da daha yüksek bir sürüm olduğunu doğrulayın.
  - Bağlantı öncesi çıkışlar yazma hakkında daha fazla bilgi için, bkz. “Bir havuzdaki bağlanma öncesi çıkışı kullanarak bağlantı tanımlarına gönderme yapılması” sayfa 404.
- Çıkışları yayınla
  - MQI ve DCI çağrılarını işaret etmek için MQPSXP yapısının **pEntryPoints** parametresini kullanın.
  - You must check that the MQPSXP version number is at version 2 or higher before using **pEntryPoints**.
  - Yayınlama çıkışlarını yazma hakkında daha fazla bilgi için bkz. “Yayınlama çıkışlarının yazılması ve derlenmesi” sayfa 406.
- Küme iş yükü çıkışları
  - MQWXP yapısının **pEntryPoints** parametresini kullanarak MQXCLWLN çağrılarını işaret edin.
  - **pEntryPoints**komutunu kullanmadan önce, MQWXP sürüm numarasının sürüm 4 ya da daha yüksek bir sürüm olduğunu doğrulayın.
  - Küme iş yükü çıkışları yazma hakkında daha fazla bilgi için, bkz. “Küme iş yükü çıkışlarının yazılması ve derlenmesi” sayfa 408.

Örneğin, bir kanal çıkışında MQPUT çağrılıyor:

```
pChannelExitParms -> pEntryPoints -> MQPUT_Call(pChannelExitParms -> Hconn,
                                                Hobj,
                                                &md,
                                                &pmo,
                                                messlen,
                                                buffer,
                                                &CompCode,
                                                &Reason);
```

"Örnek WebSphere MQ programları" sayfa 92içinde başka örnekler de görülebilir.

3. Çıkışı derleyin:
   - IBM WebSphere MQ kitaplıklarına bağlanmayın.
   - Çıkışınızdaki IBM WebSphere MQ kitaplıklarına gömülü bir RPath eklemeyin.
   - Çıkışınızı derlemeye ilişkin ek bilgi için aşağıdaki başlıklara bakın:
     - API çıkışları: "API çıkışları derleniyor" sayfa 372.
     - Kanal çıkışları, yayınlama çıkışları, Küme iş yükü çıkışları: "Windows, UNIX and Linux sistemlerinde kanal çıkış programlarının derlenmesi" sayfa 395.
     - Veri dönüştürme çıkışları: "Veri dönüştürme çıkışları yazılıyor" sayfa 397.
4. Çıkışı aşağıdaki yerlerden birine koyun:
   - Çıkışı yapılandırırken tam olarak nitelendirildiğiniz bir seçim yolu
   - Belirli bir kuruluş dizininde varsayılan çıkış yolu. Örneğin, *MQ_DATA_PATH*/exits/installation2.
   - Varsayılan çıkış yolu

     Varsayılan çıkış yolu, 32 bit çıkışlar için *MQ_DATA_PATH*/exits ve 64 bit çıkışlar için *MQ_DATA_PATH*/exits64 ' dir. Bu yolları qm.ini ya da mqclient.ini dosyasında değiştirebilirsiniz. Ek bilgi için Çıkış yolubaşlıklı konuya bakın. Windows ve Linuxişletim sistemi üzerinde, yolu değiştirmek için WebSphere MQ Explorer olanağını kullanabilirsiniz:

     a. Kuyruk yöneticisi adını sağ tıklatın
     b. **Özellikler ...**düğmesini tıklatın.
     c. **Dahili**' yi tıklatın.
     d. Çıkışlar varsayılan yolu alanında, çıkış programını tutan dizinin yol adını belirtin.

   Bir çıkış hem belirli bir kuruluş dizinine, hem de varsayılan yol dizinine yerleştirilirse, belirli bir kuruluş dizini çıkışı, yol adı verilen WebSphere MQ kuruluşu tarafından kullanılır. For example, the exit is placed in /exits/installation2 and in /exits, but not in /exits/installation1. WebSphere MQ kuruluşu installation2 , /exits/installation2' tan çıkışı kullanır. WebSphere MQ kuruluşu installation1 , /exits dizininden çıkışa kullanır.
5. Gerekiyorsa, çıkışı yapılandırın:
   - Kurulabilir hizmetler: "Hizmetlerin ve bileşenlerin yapılandırılması" sayfa 367.
   - API çıkışları: "API çıkışlarını yapılandırma" sayfa 376.
   - Kanal çıkışları: "Kanal çıkışlarının yapılandırılması" sayfa 396.
   - Çıkışlar yayınlayın: "Yayınlama çıkışlarının yapılandırılması" sayfa 407.
   - Bağlantı öncesi çıkışlar: "İstemci yapılandırma dosyasınınPreConnect kısmı" sayfa 405.

UNIX, Linux ve Windows için kurulabilir hizmetler ve bileşenler

Bu bölümde, kurulabilir hizmetler ve bunlarla ilişkili işlevler ve bileşenler tanıtılır. Bu işlevlere ilişkin arabirim, sizin ya da yazılım satıcılarının bileşen sağlayabilmesi için belgelenmiş olabilir.

WebSphere MQ kurulabilir hizmetlerini sağlamanın başlıca nedenleri şunlardır:

- WebSphere MQ ürünleri tarafından sağlanan bileşenleri kullanıp kullanmayacağınızı ya da bunları başkalarıyla değiştirmek ya da bunları başka kullanıcılarla değiştirmek için kullanacağınız esnekliği sağlamak.
- To allow vendors to participate, by providing components that might use new technologies, without making internal changes to WebSphere MQ products.
- WebSphere MQ ' un yeni teknolojileri daha hızlı ve daha ucuz bir şekilde kullanmalarına izin vermek ve ürünleri daha önce ve daha düşük fiyatlarda sağlamak.

Kurulabilir hizmetler ve *hizmet bileşenleri* , WebSphere MQ ürün yapısının bir parçasıdır. Bu yapının merkezinde, kuyruk yöneticisinin, Message Queue Interface (İleti Kuyruğu Arabirimi) ile ilişkili işlevi ve kuralları uygulayan bölümü yer alıyor. Bu merkezi parça, çalışmasını gerçekleştirmek için *kurulabilir hizmetler*adı verilen bir dizi hizmet işlevini gerektirir. Kurulabilir hizmetler şunlardır:

- Yetkilendirme hizmeti
- Ad hizmeti

Her kurulabilir hizmet, bir ya da daha fazla *hizmet bileşeni*kullanılarak uygulanan bir ilgili işlev kümesidir. Her bir bileşen, düzgün bir şekilde tasarlanmış, genel kullanıma açık bir arabirim kullanılarak çağrılır. Bu, bağımsız yazılım satıcılarının ve diğer üçüncü kişilerin WebSphere MQ ürünleri tarafından sağlananları büyütmek ya da değiştirmek için kurulabilir bileşenler sağlamasına olanak sağlar. Çizelge 52 sayfa 360 , kullanılabilecek hizmetleri ve bileşenleri özetler.

Çizelge 52. Kurulabilir hizmet bileşenleri özeti

Kurulabilir hizmet	Sağlanan bileşen	İşlev	Gereksinmeler
Yetkilendirme hizmeti	nesne yetkisi yöneticisi (OAM)	Komutlara ve MQI çağrılarına ilişkin yetki denetimi sağlar. Kullanıcılar OAM ' yi büyütmek ya da değiştirmek için kendi bileşenlerinden yazabilir. Örneğin, bir kullanıcı kimliğinin kuyruğu açma yetkisi olup olmadığını denetlemek için.	(Uygun platform yetkilendirme olanakları varsayılır)
Ad hizmeti	Yok	Kuyruk yöneticisine, belirlenen bir kuyruğa sahip olan kuyruk yöneticisinin adını aramak için destek sağlar. • Kullanıcı tanımlı **Not:** Paylaşılan kuyrukların CELL olarak ayarlanmış *Scope* özniteliği olmalıdır.	• Üçüncü taraf ya da kullanıcı tarafından yazılan bir ad yöneticisi

Kurulabilir hizmetler arabirimi, Kurulabilir hizmetler arabirimi başvuru bilgileriiçinde açıklanmıştır.

Hizmet bileşeni yazılması

Bu bölümde hizmetler, bileşenler, giriş noktaları ve dönüş kodları arasındaki ilişki anlatılır.

İşlevler ve bileşenler

Her hizmet, bir dizi ilgili işlevden oluşur. Örneğin, ad hizmeti aşağıdakiler için işlev içerir:

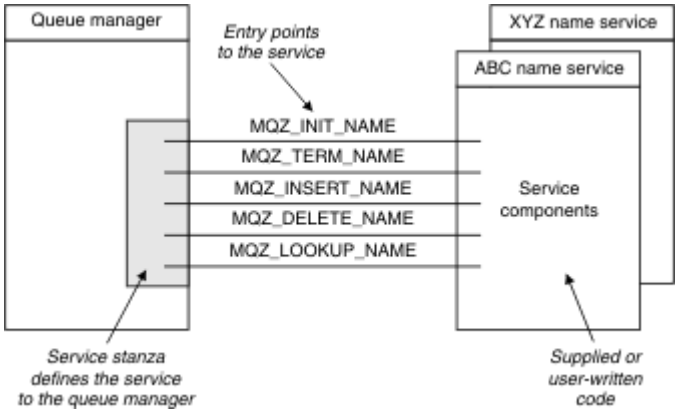
- Bir kuyruk adı aranır ve kuyruğun tanımlandığı kuyruk yöneticisinin adını döndürür.
- Hizmet dizinine kuyruk adı eklenmesi
- Bir kuyruk adının hizmet dizininden silinmesi

Ayrıca, başlatma ve sonlandırma işlevlerini de içerir.

Kurulabilir bir hizmet, bir ya da daha çok hizmet bileşeni tarafından sağlanır. Her bileşen, söz edilen hizmet için tanımlanmış işlevlerin bazılarını ya da tümünü gerçekleştirebilir. Örneğin, AIXiçin WebSphere MQ ' ta sağlanan yetkilendirme hizmeti bileşeni OAM, kullanılabilir tüm işlevleri gerçekleştirir. Ek bilgi için “Yetkilendirme hizmeti arabirimi” sayfa 364 başlıklı konuya bakın. Bu bileşen ayrıca, hizmetin uygulanması için gereken temel kaynakları ya da yazılımları (örneğin, LDAP dizini) yönetmekten de sorumludur. Yapılandırma dosyaları, bileşeni yüklemek ve sağladığı işlevsel yordamların adreslerini belirlemek için standart bir yol sağlar.

Şekil 72 sayfa 361 , hizmetlerin ve bileşenlerin nasıl ilgili olduğunu gösterir:

- Bir hizmet, bir yapılandırma dosyasındaki stanzas tarafından kuyruk yöneticisinde tanımlanır.
- Her hizmet, kuyruk yöneticisinde sağlanan kodla desteklenir. Kullanıcılar bu kodu değiştiremez ve bu nedenle kendi hizmetlerini oluşturamaz.
- Her hizmet bir ya da daha fazla bileşen tarafından uygulanır; bunlar ürünle birlikte ya da kullanıcı tarafından yazılmış olabilir. Bir hizmet için birden çok bileşen çağrılabilir, her biri hizmet içinde farklı tesisleri destekleyebilir.
- Giriş noktaları, hizmet bileşenlerini kuyruk yöneticisinde destekleyici kodlara bağlar.



Şekil 72. Hizmetlerin, bileşenlerin ve giriş noktalarının anlaşılması

Giriş noktaları

Her hizmet bileşeni, belirli bir kurulabilir hizmeti destekleyen yordamların giriş noktası adreslerinin bir listeiyle gösterilir. Kurulabilir hizmet, her yordam tarafından gerçekleştirilecek işlevi tanımlar.

Hizmet bileşenlerinin yapılandırıldığı sırada sipariş edilmesi, hizmete ilişkin bir isteği karşılamaya yönelik giriş-noktaların çağrıldığı sırayı tanımlar.

Sağlanan `cmqzc.`hüstbilgi dosyasında, sağlanan giriş noktalarının her bir hizmete ilişkin bir MQZID_ öneki var.

Hizmetler mevcutsa, hizmetler önceden tanımlanmış bir sırayla yüklenir. Aşağıdaki listede hizmetler ve bunların kullanıma hazırlandıkları sıra gösterilir.

1. `NameService`
2. `AuthorizationService`
3. `UserIdentifierService`

`AuthorizationService` , varsayılan olarak yapılandırılan tek hizmettir. `NameService` ve `UserIdentifierService` ' yi kullanmak istiyorsanız el ile yapılandırın.

Hizmetler ve hizmet bileşenleri bire bir ya da bire bir eşlemeye sahiptir. Her hizmet için birden çok hizmet bileşeni tanımlanabilir. On UNIX and Linux systems, the `ServiceComponent` stanza's Service value must match the Service stanza's Name value in the qm.ini file. Windowsüzerinde, `ServiceComponent`' in Hizmet kaydı anahtar değeri, Ad kayıt dosyası anahtar değeriyle eşleşmelidir; şu şekilde tanımlanır:

HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries\CurrentVersion\Configuration\QueueManager*qmname*\ ; burada *qmname* , kuyruk yöneticisinin adıdır.

For UNIX and Linux systems, service components are started in the order they are defined in the qm.ini file. On Pencereler, because the Windows registry is used, WebSphere MQ issues a **RegEnumKey** call which returns the values in alphabetic order. Bu nedenle, Pencereler üzerinde hizmetler, kayıtta tanımlandıkça, alfabetik sırayla çağrılır.

ServiceComponent tanımlarının sıralaması önemlidir. Bu sıralama, belirli bir hizmet için bileşenlerin çalıştırılmasına ilişkin sırayı belirler. Örneğin, Windows üzerindeki AuthorizationService , MQSeries.WindowsNT.auth.serviceadlı varsayılan OAM bileşeniyle yapılandırılır. Varsayılan OAM ' yi geçersiz kılmak için bu hizmet için ek bileşenler tanımlanabilir. MQCACF_SERVICE_COMPONENT belirtilmediyse, isteği işlemek için alfabetik sırada karşılaşılan ilk bileşen kullanılır ve ilgili bileşenin adı kullanılır.

Dönüş kodları

Hizmet bileşenleri, çeşitli koşullarla ilgili raporlama yapmak üzere kuyruk yöneticisine dönüş kodları sağlar. İşlemlerin başarılı ya da başarısız olduğunu bildirirler ve kuyruk yöneticisinin bir sonraki hizmet bileşenine ilerleyip ilerlemeyeceğini belirtir. Ayrı bir *Devamı* parametresi bu göstergeyi taşır.

Bileşen verileri

Tek bir hizmet bileşeni, verilerin çeşitli işlevleri arasında paylaşılmasını gerektirebilir. Kurulabilir hizmetler, bir hizmet bileşeninin her çağrısına geçirilmek üzere isteğe bağlı bir veri alanı sağlar. Bu veri alanı, hizmet bileşeninin dışlayıcı kullanımı içindir. Bu, farklı adres alanlarından ya da süreçlerden yapılmış olsa da, belirli bir işlevin tüm çağrıları tarafından paylaşılır. Her çağrıldığında, hizmet bileşeninden adreslenebilir bir şekilde verileceği garanti edilir. Bu alanın boyutunu *ServiceComponent* stanza içinde bildirmeniz gerekir.

Bileşenlerin başlatılması ve sona erdirilmesi
Bileşen kullanıma hazırlama ve sonlandırma seçeneklerinin kullanımı.

Bileşen kullanıma hazırlama yordamı çağrıldığında, bileşen tarafından desteklenen her bir giriş noktası için kuyruk yöneticisi **MQZEP** işlevini çağırmalıdır. **MQZEP** , hizmete ilişkin bir giriş noktasını tanımlar. Tanımlanmamış tüm çıkış noktalarının boş olduğu varsayılır.

Bir bileşen, her zaman birincil kullanıma hazırlama seçeneğiyle bir kez çağrılır; bu seçenek başka bir şekilde çağrılmadan önce çağrılır.

Bazı altyapılarda ikincil kullanıma hazırlama seçeneğiyle bir bileşen çağrılabilir. Örneğin, hizmete erişildiği her işletim sistemi işlemi, iş parçacığı ya da görev için bir kez çağrılabilir.

İkincil başlatma kullanılırsa:

- İkincil kullanıma hazırlama işlemi için bileşen birden çok kez çağrılabilir. Bu tür her çağrı için, hizmet artık gerekmediği durumlarda ikincil sona erdirme için eşleşen bir çağrı yayınlanır.

  Adlandırma hizmetleri için bu, MQZ_TERM_NAME çağrısıdır.

  Yetki hizmetleri için bu, MQZ_TERM_AUTHORITY çağrısıdır.
- Bileşen birincil ve ikincil kullanıma hazırlama işlemi için her çağrıldığında, giriş noktalarının yeniden belirtilmesi gerekir (MQZEP çağrılarak).
- Bileşen verilerinin yalnızca bir kopyası bileşen için kullanılır; her ikincil kullanıma hazırlama işlemi için farklı bir kopya yoktur.
- İkincil kullanıma hazırlama gerçekleştirilmeden önce, hizmetin (işletim sistemi işleminden, iş parçacığısından ya da görevden uygun olduğu şekilde) başka çağrılar için çağrılmaz.
- The component must set the *Version* parameter to the same value for primary and secondary initialization.

Bileşen her zaman birincil sonlandırma seçeneği ile birlikte çağrılır ve bu seçenek artık gerekli değildir. Bu bileşen için başka arama yapılmayacak.

İkincil kullanıma hazırlama işlemi için çağrıldıysa, bileşen ikincil sonlandırma seçeneğiyle birlikte çağrılır.

Nesne yetkisi yöneticisi (OAM)
WebSphere MQ ürünleriyle birlikte sağlanan yetkilendirme hizmeti bileşeni, Object Authority Manager (OAM) adı verilir.

Varsayılan değer olarak, OAM etkindir ve **dspmqaut** (görüntüleme yetkilisi),**dmpmqaut** (döküm yetkisi) ve **setmqaut** (set ya da reset authority) denetim komutlarıyla çalışır.

Bu komutların sözdizimi ve kullanım şekli Denetim komutlarıiçinde açıklanmıştır.

OAM, bir birincil kullanıcı ya da grubun *varlığı* ile çalışır.

- UNIX and Linux sistemlerinde:
  - Asıl ad, bir kullanıcı kimliği ya da kullanıcı adına çalışan bir uygulama programıyla ilişkilendirilmiş bir kimlikle ilişkilendirilir.
  - the group is a UNIX or Linux system-defined collection of principals.
  - Yetkilendirmeler yalnızca grup düzeyinde verilebilir ya da iptal edilebilir. Bir kullanıcının yetkisini verme ya da bu yetkiyi iptal etme isteği, o kullanıcıya ilişkin birincil grubu günceller.
- Windows sistemlerinde:
  - Asıl ad, bir Windows kullanıcı kimliğidir ya da bir kullanıcı adına çalışan bir uygulama programıyla ilişkilendirilmiş bir tanıtıcıdır.
  - Grup, bir Windows grudur.
  - Yetkilendirmeler, birincil kullanıcı ya da grup düzeyinde verilebilir ya da iptal edilebilir.

Bir MQI isteği yapıldığında ya da bir komut verildiğinde OAM, işlemin yapabildiğini görmek için işlemle ilişkili varlığın yetkilendirmesini denetler:

- İstenen işlemi gerçekleştirin.
- Belirtilen kuyruk yöneticisi kaynaklarına erişin.

Yetkilendirme hizmeti, kendi yetkilendirme hizmeti bileşeninizi yazarak kuyruk yöneticileri için sağlanan yetki denetimini artırmanızı ya da değiştirmenizi sağlar.

Ad hizmeti
Ad hizmeti, belirlenen bir kuyruğa sahip olan kuyruk yöneticisinin adını aramak için kuyruk yöneticisine destek sağlayan kurulabilir bir hizmettir. Bir ad hizmetinden başka bir kuyruk özniteliği alınamıyor.

Ad hizmeti, bir uygulamanın çıkış için yerel kuyruklar gibi uzak kuyruklar açmasını sağlar. Kuyruklar dışındaki nesneler için bir ad hizmeti çağrılmaz.

Not: The remote queues ***gerekir*** have their *Scope* attribute set to CELL.

Bir uygulama bir kuyruğu açtığında, kuyruk yöneticisinin dizininde ilk olarak kuyruğun adını arar. Burada bulamazsa, kuyruk adını tanıyan birini buluncaya kadar, yapılandırılmış olduğu kadar çok sayıda ad hizmeti yapılandırılmıştır. Ad tanınmazsa, açma işlemi başarısız olur.

Ad hizmeti, o kuyruk için sahip olan kuyruk yöneticisini döndürür. Daha sonra kuyruk yöneticisi, özgün istekte kuyruk ve kuyruk yöneticisi adını belirtmiş gibi MQOPER isteğiyle devam eder.

Ad hizmeti arabirimi (NSI), WebSphere MQ çerçevesinin bir parçasıdır.

Ad hizmeti nasıl çalışır

If a queue definition specifies the *Scope* attribute as queue manager, that is, SCOPE(QMGR) in MQSC, the queue definition (along with all the queue attributes) is stored in the queue manager's directory only. Bu, kurulabilir bir hizmetle değiştirilemez.

If a queue definition specifies the *Scope* attribute as cell, that is, SCOPE(CELL) in MQSC, the queue definition is again stored in the queue manager's directory, along with all the queue attributes. Ancak, kuyruk ve kuyruk yöneticisi adı aynı zamanda bir ad hizmetinde de saklanır. Bu bilgileri saklayabilen bir hizmet yoksa, *Scope* hücresine sahip bir kuyruk tanımlanamaz.

Bilgilerin saklanabileceği dizin, hizmet tarafından yönetilebilir ya da hizmet, bu amaçla temel bir hizmeti (örneğin, bir LDAP dizini) kullanabilir. Her iki durumda da, bileşen ve kuyruk yöneticisi belirtik olarak silininceye kadar, dizinde saklanan tanımların kalıcı olarak saklanmaması gerekir.

Not:

1. Uzak bir anasistemin yerel kuyruk tanımlamasına (CELL kapsamı ile) bir adlandırma dizini hücresindeki farklı bir kuyruk yöneticisiyle ileti göndermek için, bir kanal tanımlamanız gerekir.
2. CELL kapsamına girse bile, doğrudan uzak kuyruktan ileti alamazsınız.
3. CELL kapsamı içeren bir kuyruğa gönderilirken uzak kuyruk tanımlaması gerekmez.
4. Hedef kuyruk yöneticisi ve bir çift kanal tanımlaması için hala bir iletim kuyruğuna gereksinim duyarsanız, adlandırma hizmeti merkezi olarak hedef kuyruğu tanımlar. Bunun yanı sıra, yerel sistemdeki iletim kuyruğu, hedef kuyruğu bulunduran kuyruk yöneticisiyle aynı adı ve uzak sistemde hücre kapsamı ile aynı adı olmalıdır.

   For example, if the remote queue manager has the name QM01, the transmission queue on the local system must also have the name QM01.

Yetkilendirme hizmeti arabirimi
Yetki hizmeti, kuyruk yöneticisi tarafından kullanılmak üzere giriş noktaları sağlar.

Giriş noktaları şunlardır:

MQZ_AUTHENTICATE_USER
Kullanıcı kimliği ve parola doğrulanır ve kimlik bağlamı alanları ayarlayabilir.

MQZ_CHECK_AUTHORITY
Bir varlığın belirtilen bir nesne üzerinde bir ya da daha fazla işlem gerçekleştirme yetkisinin olup olmadığını denetler.

MQZ_CHECK_IMTIYAZLI
Belirtilen kullanıcının ayrıcalıklı bir kullanıcı olup olmadığını denetler.

MQZ_COPY_ALL_AUTHORITY
Başvurulan bir nesne için var olan tüm geçerli yetkileri başka bir nesneye kopyalar.

MQZ_DELETE_AUTHORITY
Belirtilen nesneyle ilişkili tüm yetkileri siler.

MQZ_ENUMERATE_AUTHORITY_DATA
Belirtilen seçim ölçütleriyle eşleşen tüm yetki verilerini alır.

MQZ_FREE_USER
Ayrılmış kaynak ayrılmış kaynakları boşaltabiliyor.

MQZ_GET_AUTHORITY
Bir varlığın belirtilen bir nesneye erişmesi için sahip olduğu yetkiyi alır.

MQZ_GET_AÇIKLANAMAZ_YETKISI
Adlandırılmış bir grubun belirli bir nesneye (**Kimse** grubu ek yetkisi olmadan) erişmesi ya da belirtilen birincil kullanıcının birincil grubunun belirtilen bir nesneye erişmek zorunda olduğu yetkiye sahip olması ya da yetkisi alır.

MQZ_INIT_AUTHORITY
Yetkilendirme hizmeti bileşenini kullanıma hazırlar.

MQZ_SORGULAMA
Yetkilendirme hizmetinin desteklenen işlevselliğini sorgular.

MQZ_REFRESH_CACHE
Tüm yetkileri yenileyin.

MQZ_SET_AUTHORITY
Bir varlığın belirli bir nesneye sahip olduğu yetkiyi ayarlar.

MQZ_TERM_AUTHORITY
Yetkilendirme hizmeti bileşenini sona erdirir.

In addition, on WebSphere MQ for Pencereler, the authorization service provides the following entry points for use by the queue manager:

- **MQZ_CHECK_AUTHORITY_2**
- **MQZ_GET_AUTHORITY_2**
- **MQZ_GET_EXPLICIT_AUTHORITY_2**
- **MQZ_SET_AUTHORITY_2**

Bu giriş noktaları, Windows Security Identifier (NT SID) ' nin kullanımını destekler.

Bu adlar, bileşen işlevlerinin prototipini oluşturmak için kullanılabilen `cmqzc.` hüstbilgi dosyasında **tipdef**olarak tanımlanır.

Kullanıma hazırlama işlevi (**MQZ_INIT_AUTHORITY**), bileşenin ana giriş noktalarından biri olmalıdır. Diğer işlevler, başlatma işlevinin bileşen giriş noktası vektörine eklediği giriş noktası adresinden çağrılır.

Ad hizmeti arabirimi
Ad hizmeti, kuyruk yöneticisi tarafından kullanılmak üzere giriş noktaları sağlar.

Aşağıdaki giriş noktaları sağlanmıştır:

MQZ_INIT_NAME
Ad hizmeti bileşenini başlatın.

MQZ_TERM_ADı
Ad hizmeti bileşenini sona erdirin.

MQZ_LOOKUP_NAME
Belirtilen kuyruk için kuyruk yöneticisi adına bakın.

MQZ_INSERT_NAME
Belirtilen kuyruk için sahip olan kuyruk yöneticisi adını içeren bir giriş, hizmet tarafından kullanılan dizine ekler.

MQZ_DELETE_NAME
Belirlenen kuyruğa ilişkin girişi, hizmet tarafından kullanılan dizinden silin.

Yapılandırılmış birden fazla ad hizmeti varsa:

- Arama için, kuyruk adı çözülünceye kadar (herhangi bir bileşen aramanın durması gerektiğini belirtmedikçe), listedeki her hizmet için MQZ_LOOKUP_NAME işlevi çağrılır.
- Araya ekleme için, bu işlevi destekleyen listedeki ilk hizmet için MQZ_INSERT_NAME işlevi çağrılır.
- Silme işlemi için, bu işlevi destekleyen listedeki ilk hizmet için MQZ_DELETE_NAME işlevi çağrılır.

Ekleme ve silme işlevlerini destekleyen birden çok bileşende bulunmayın. Ancak, yalnızca aramanın desteklediği bir bileşen uygulanabilir ve örneğin, listedeki diğer herhangi bir ad hizmeti bileşeni tarafından adının tanımlanabileceği bir kuyruk yöneticisine herhangi bir ad tarafından tanınmayan herhangi bir adı çözmek için listedeki son bileşen olarak kullanılabilir.

C programlama dilinde, adlar, tipdef deyimi kullanılarak işlev veri tipleri olarak tanımlanır. Bu bilgiler, parametrelerin doğru olduğundan emin olmak için hizmet işlevlerinin prototipini oluşturmak için kullanılabilir.

Kurulabilir hizmetlere özgü tüm malzemeyi içeren üstbilgi dosyası C dili için `cmqzc.h` ' dir.

Bileşenin ana giriş noktası olması gereken, kullanıma hazırlama işlevinin (MQZ_INIT_NAME) dışında, işlevlerin başlatılması, MQZEP çağrısını kullanarak, kullanıma hazırlama işlevinin eklediği giriş noktası adresi tarafından çağrılır.

Birden çok hizmet bileşeninin kullanılması
Bir hizmet için birden çok bileşen kurabilirsiniz. Bu, bileşenlerin yalnızca hizmetin kısmi somutlamalarını sağlamasına ve kalan işlevleri sağlamak için diğer bileşenlere güvenmesine olanak sağlar.

Birden çok bileşeni kullanma örneği

Suppose you create two a name services components called ABC_name_serv and XYZ_name_serv.

ABC_name_serv
Bu bileşen, hizmet dizininden ad eklemeyi ya da bir adı silmesini destekler, ancak kuyruk adını aramaktan destek olmaz.

XYZ_name_serv
Bu bileşen, bir kuyruk adını bakmayı destekler, ancak hizmet dizininden bir adı eklemeyi ya da bir adı silmeyi desteklemez.

ABC_name_serv bileşeni kuyruk adlarının bir veritabanını bulundurur ve hizmet dizininden bir ad eklemek ya da silmek için iki basit algoritma kullanır.

XYZ_name_serv bileşeni, çağrıldığı herhangi bir kuyruk adı için sabit bir kuyruk yöneticisi adı döndüren basit bir algoritma kullanır. Kuyruk adları veritabanı tutmaz ve bu nedenle araya ekleme ve silme işlevlerini desteklemez.

Bileşenler aynı kuyruk yöneticisine kurulur. The *ServiceComponent* stanzas are ordered so that component ABC_name_serv is invoked first. Bir bileşen dizinine kuyruk ekleme ya da silme çağrıları, ABC_name_servbileşeni tarafından işlenir; bu işlevleri gerçekleştiren tek bir bileşen vardır. However, a lookup call that component ABC_name_serv cannot resolve is passed on to the lookup-only component, XYZ_name_serv. Bu bileşen, basit algoritmasından bir kuyruk yöneticisi adı sağlar.

Birden çok bileşen kullanılırken giriş noktalarının atlanması

Bir hizmet sağlamak için birden çok bileşen kullanmaya karar verirseniz, belirli işlevleri gerçekleştirmeyen bir hizmet bileşeni tasarlayabilirsiniz. Kurulabilir hizmetler çerçevesi, atlayabileceğiniz herhangi bir kısıtlama içermiyor. Ancak, belirli kurulabilir hizmetler için, bir ya da daha çok işlevin eksik olması, hizmetin amacı ile mantıksal olarak tutarsız olabilir.

Birden çok bileşenle kullanılan giriş noktaları örneği

Çizelge 53 sayfa 366 içinde, iki bileşenin kurulu olduğu kurulabilir ad hizmetine bir örnek gösterilmektedir. Her biri, bu kurulabilir hizmetle ilişkilendirilmiş farklı bir işlev kümesini destekler. Araya ekleme işlevi için, ilk olarak ABC bileşeni giriş noktası çağrılır. Hizmet için tanımlanmamış giriş noktaları (**MQZEP**kullanılarak) NULL olduğu varsayılır. Çizelgede kullanıma hazırlama noktasına ilişkin bir giriş noktası sağlanmıştır; ancak, kullanıma hazırlama, bileşenin ana giriş noktası tarafından gerçekleştirildiğinden, bu gerekli değildir.

Kuyruk yöneticisi kurulabilir bir hizmet kullanmak zorunda olduğunda, o hizmet için tanımlanan giriş noktalarını kullanır (Çizelge 53 sayfa 366' taki sütunlar). Kuyruk yöneticisi, her bileşeni sırayla almak için gereken işlevi gerçekleştiren yordamın adresini belirler. Daha sonra, varsa, yordamı çağırır. İşlem başarılı olursa, herhangi bir sonuç ve durum bilgisi kuyruk yöneticisi tarafından kullanılır.

Çizelge 53. Kurulabilir bir hizmete ilişkin giriş noktaları örneği

İşlev numarası	**ABC ad hizmeti bileşeni**	**XYZ adı hizmet bileşeni**
MQZID_INIT_NAME (Kullanıma Hazırla)	ABC_initialize ()	XYZ_initialize ()
MQZID_TERM_NAME (Sonlandır)	ABC_terminate ()	XYZ_terminate ()
MQZID_INSERT_NAME (Ekle)	ABC_Insert ()	BOŞ DEĞERLİ
MQZID_DELETE_NAME (Sil)	ABC_Delete ()	BOŞ DEĞERLİ

Çizelge 53. Kurulabilir bir hizmete ilişkin giriş noktaları örneği (devamı var)		
İşlev numarası	**ABC ad hizmeti bileşeni**	**XYZ adı hizmet bileşeni**
MQZID_LOOKUP_NAME (Arama)	BOŞ DEĞERLİ	XYZ_Lookup ()

Yordam yoksa, kuyruk yöneticisi bu işlemi listede sonraki bileşen için yineler. Ayrıca, yordam varsa, ancak işlemi gerçekleştiremediğini belirten bir kod döndürürse, girişim sonraki kullanılabilir bileşenle devam eder. Hizmet bileşenlerindeki yordamlar, işlemi gerçekleştirmek için başka bir girişimde bulunmayacağına işaret eden bir kod döndürebilir.

Hizmetlerin ve bileşenlerin yapılandırılması

Configure service components using the queue manager configuration files, except on Pencereler systems, where each queue manager has its own stanza in the Registry.

1. Kuyruk yöneticisine hizmet tanımlamak ve modülün yerini belirtmek için, kuyruk yöneticisi yapılanış kütüğüne stanzas ekleyin.

   Kullanılan her hizmetin, kuyruk yöneticisine hizmet tanımlayan bir *Service* stanza olmalıdır.

   Bir hizmet içindeki her bir bileşen için bir *ServiceComponent* stanza olmalıdır. Bileşene ilişkin kodu içeren modülün adını ve yolunu belirtir.

   Daha fazla bilgi için bkz. “Hizmet stanza biçimi” sayfa 367 ve “Hizmet bileşeni stanza biçimi” sayfa 368

   Object Authority Manager (OAM) olarak bilinen yetkilendirme hizmeti bileşeni ürünle birlikte sağlanır. Bir kuyruk yöneticisi yarattığınızda, kuyruk yöneticisi yapılanış kütüğü (ya da Windows sistemlerindeki kayıt dosyası), yetki hizmetine ilişkin uygun kısmı ve varsayılan bileşen (OAM) için uygun taneleri içerecek şekilde otomatik olarak güncellenir. Diğer bileşenler için, kuyruk yöneticisi yapılanış kütüğünü el ile yapılandırmanız gerekir.

   Kuyruk yöneticisi başlatıldığında, dinamik bağ tanımı kullanılarak, bu altyapıda desteklendiği durumlarda, her hizmet bileşenine ilişkin kod kuyruk yöneticisine yüklenir.
2. Bileşeni etkinleştirmek için kuyruk yöneticisini durdurup yeniden başlatın.

Hizmet stanza biçimi

Hizmet kısmı, hizmetin adını ve hizmet için tanımlanan giriş noktalarının sayısını içerir.

stanza ' nın formatı şu şekilde:

```
Service:
  Name=<service_name>
  EntryPoints=<entries>
```

Burada:

<service_name>
Hizmetin adı Bu, hizmet tarafından tanımlanır.

<entries>
Hizmet için tanımlanan giriş noktalarının sayısı. Bu, kullanıma hazırlama ve sonlandırma giriş noktalarını içerir.

Windows sistemleri için hizmet stanza biçimi

Windows sistemlerinde, *Service* Stanza bir `SecurityPolicy` özniteliğini içerir.

stanza ' nın formatı şöyle:

```
Service:
  Name=<service_name>
  EntryPoints=<entries>
  SecurityPolicy=<policy>
```

Burada:

<service_name>
Hizmetin adı Bu, hizmet tarafından tanımlanır.

<entries>
Hizmet için tanımlanan giriş noktalarının sayısı. Bu, kullanıma hazırlama ve sonlandırma giriş noktalarını içerir.

<policy>
NTSIDsRequired (Windows Güvenlik Tanıtıcısı) ya da Default. NTSIDsRequiredbelirtmezseniz, Default değeri kullanılır. This attribute is valid only if Name has a value of AuthorizationService.

Ayrıca bkz. "Yetkilendirme hizmeti stanzaları yapılandırılıyor: Windows sistemleri" sayfa 369.

Hizmet bileşeni stanza biçimi

Hizmet bileşeni kısmına ilişkin biçim şöyledir:

```
ServiceComponent:
  Service=<service_name>
  Name=<component_name>
  Module=<module_name>
  ComponentDataSize=<size>
```

Burada:

<service_name>
Hizmetin adı Bu, hizmet stanzasında belirtilen Name ile eşleşmelidir.

<component_name>
Hizmet bileşeninin açıklayıcı bir adı. Bu benzersiz olmalı ve yalnızca WebSphere MQ nesneleri (örneğin, kuyruk adları) adları için geçerli olan karakterleri içermelidir. Bu ad, hizmet tarafından oluşturulan işletmen iletilerinde ortaya çıkar. Şirket ticari markası ya da benzeri ayırt edici dizgisiyle başlayan bir ad kullanmanızı öneririz.

<module_name>
Bu bileşene ilişkin kodu içerecek modülün adı.

<size>
Her çağrışımda bileşene geçirilen bileşen verileri alanının bayt cinsinden boyutu. Bileşen verisi gerekmiyorsa sıfır değerini belirtin.

bu iki stanzalar herhangi bir sırayla ortaya çıkabilir ve bunların altındaki stanza anahtarları da herhangi bir sırada gerçekleşebilir. Bu stanzalardan herhangi biri için tüm stanza anahtarlarının mevcut olması gerekir. Bir stanza anahtarı yinelenirse, sonuncunda kullanılır.

Başlatma sırasında, kuyruk yöneticisi her bir hizmet bileşeni girişini, yapılandırma dosyasındaki sırayla işler. Daha sonra, belirtilen bileşen modülünü yükler; bileşenin giriş noktasını (bileşenin kullanıma hazırlanması için giriş noktası olmalıdır) çağırarak, bir yapılandırma tanıtıcısı iletir.

Yetkilendirme hizmeti stanzaları yapılandırılıyor: UNIX and Linux sistemleri
UNIX and Linux sistemlerinde, her kuyruk yöneticisinin kendi kuyruk yöneticisi yapılanış kütüğü vardır.

Örneğin, kuyruk yöneticisi QMNAME için kuyruk yöneticisi yapılanış kütüğünün varsayılan yolu ve kütük adı /var/mqm/qmgrs/QMNAME/qm.iniolur.

The *Service* stanza and the *ServiceComponent* stanza for the default authorization component are added to qm.ini automatically, but can be overridden by mqsnoaut. Diğer *ServiceComponent* stanzaları el ile eklenmelidir.

Örneğin, kuyruk yöneticisi yapılanış dosyasındaki şu stanzas, AIXiçin WebSphere MQ üzerinde iki yetki hizmeti bileşeni tanımlıyor. *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

```
Service:
   Name=AuthorizationService
   EntryPoints=13

ServiceComponent:
   Service=AuthorizationService
   Name=MQSeries.UNIX.auth.service
   Module=MQ_INSTALLATION_PATH/lib/amqzfu
   ComponentDataSize=0

ServiceComponent:
   Service=AuthorizationService
   Name=user.defined.authorization.service
   Module=/usr/bin/udas01
   ComponentDataSize=96
```

Şekil 73. qm.iniiçindekiUNIX and Linux yetkilendirme hizmeti dayanakları

Hizmet bileşeni kısmı (`MQSeries.UNIX.auth.service`), varsayılan yetkilendirme hizmeti bileşenini, OAM ' yi tanımlar. Bu stanza 'yı kaldırırsanız ve kuyruk yöneticisini yeniden başladıysanız, OAM devre dışı bırakılır ve yetki denetimi yapılmamaktadır.

Yetkilendirme hizmeti stanzaları yapılandırılıyor: Windows sistemleri
WebSphere MQ ' da, Windows için her kuyruk yöneticisinin kayıt defterinde kendi stanzası vardır.

The *`Service`* stanza and the *`ServiceComponent`* stanza for the default authorization component are added to the Registry automatically, but can be overridden using `mqsnoaut`. Diğer *`ServiceComponent`* stanzaları el ile eklenmelidir.

Ayrıca, `SecurityPolicy` özniteliğini, WebSphere MQ hizmetlerini kullanarak da ekleyebilirsiniz. `SsecurityPolicy` özniteliği, yalnızca *`Service`* stanza üzerinde belirtilen hizmet yetkilendirme hizmetiysa, yani varsayılan OAM ' dir. `SecurityPolicy` özniteliği, her kuyruk yöneticisi için güvenlik ilkesini belirtmenizi sağlar. Olası değerler şunlardır:

`Default`
Varsayılan güvenlik ilkesinin yürürlüğe girmesi için `Default` değerini belirtin. Bir Windows güvenlik tanıtıcısı (NT SID) belirli bir kullanıcı kimliği için OAM 'a geçirilmezse, ilgili güvenlik veritabanlarında arama yaparak uygun SID ' yi elde etmek için bir girişimde bulunmanız gerekir.

`NTSIDsRequired`
Güvenlik denetimleri gerçekleştirilirken bir NT SID 'nin OAM' ye iletilmesini gerektirir.

Hizmet stanza biçimi hakkında bilgi için bkz. “ Windows sistemleri için hizmet stanza biçimi” sayfa 367. Güvenliğe ilişkin daha fazla genel bilgi için bkz. Pencereler, UNIX and Linux sistemleri üzerinde güvenliğin ayarlanması.

Hizmet bileşeni kısmı (`MQSeries.WindowsNT.auth.service`), varsayılan yetkilendirme hizmeti bileşenini, OAM ' yi tanımlar. Bu stanza 'yı kaldırırsanız ve kuyruk yöneticisini yeniden başladıysanız, OAM devre dışı bırakılır ve yetki denetimi yapılmamaktadır.

Ad hizmeti stanzaları yapılandırılıyor: Unix ve Linux sistemleri
Kısa açıklamanızı buraya koyun; ilk paragraf ve özet için kullanılır.

Ad hizmeti için aşağıdaki UNIX and Linux yapılandırma dosyası stanzaları örnekleri, (kurgusal) ABC şirketi tarafından sağlanan bir ad hizmeti bileşeni belirtmektedir.

```
# Stanza for name service
 Service:
    Name=NameService
    EntryPoints=5

# Stanza for name service component, provided by ABC
 ServiceComponent:
    Service=NameService
    Name=ABC.Name.Service
    Module=/usr/lib/abcname
    ComponentDataSize=1024
```

Şekil 74. qm.ini içindeki ad hizmeti stanzaları (UNIX and Linux sistemleri için)

Not: Windows sistemlerinde, ad hizmeti stanza bilgileri Kayıt Defterinde saklanır.

Bir kullanıcının yetkisini değiştirdikten sonra OAM yenileniyor
WebSphere MQ' da, bir kullanıcının yetki grubu üyeliğini değiştirdikten hemen sonra, kuyruk yöneticisini durdurup yeniden başlatmak gerekmeden işletim sistemi düzeyinde yapılan değişiklikleri yansıtarak OAM yetki grubu bilgilerini yenileyebilirsiniz. Bunu yapmak için **REFRESH SECURITY** komutunu verin.

Not: Yetkileri `setmqaut` komutuyla değiştirdiğinizde, OAM hemen bu tür değişiklikleri uygular.

Kuyruk yöneticileri, yetki verilerini SYSTEM.AUTH.DATA.QUEUE. Bu veriler `amqzfuma.exe` tarafından yönetilir.

İlgili başvurular
Güvenliği yenileme

API çıkışlarının yazılması ve derlenmesi

API çıkışları, WebSphere MQ API çağrılarının (MQPUT ve MQGET gibi) davranışını değiştiren kod yazmanızı sağlar ve bu çağrıların hemen ardından ya da hemen sonra bu kodu eklemenize olanak sağlar.

Not: Not supported on WebSphere MQ for z/OS.

API çıkışları neden kullanılır?

Uygulamalarınızın her birinin yapması gereken belirli bir işi vardır ve bu işin kodu, görevi mümkün olduğunca verimli bir şekilde yapmalıdır. At a higher level, you might want to apply standards or business processes to a particular queue manager for **Tümü** the applications that use that queue manager. Bunu tek tek uygulamalar düzeyinin üzerinde yapmak daha verimli ve etkilenen her uygulamanın kodunu değiştirmek zorunda kalmaksızın.

API çıkışlarının yararlı olabileceği alanlar için birkaç öneri vardır:

- *güvenlik*için, uygulamaların bir kuyruğa ya da kuyruk yöneticisine erişim yetkisi olup olmadığını denetleyerek kimlik doğrulaması sağlayabilirsiniz. Ayrıca, polis uygulamalarının API ' yı kullanarak, bireysel API çağrılarını doğrulayabilir, hatta kullandıkları parametreleri de doğrulayabilirsiniz.
- *esneklik*için, bu ortamdaki verilere dayalı olan uygulamaları değiştirmeden, iş ortamınızdaki hızlı değişikliklere yanıt verebilirsiniz. Örneğin, faiz oranlarındaki değişikliklere, para birimi döviz kurlarına ya da bir üretim ortamındaki bileşenlerin fiyatlarına yanıt veren API çıkışlarına sahip olabilir.
- Bir kuyruk ya da kuyruk yöneticisinin *izleme* kullanımı için, uygulama ve ileti akışını izleyebilirsiniz, API çağrılarında hataları günlüğe kaydedebilir, muhasebe işlemleri için denetleme izlerini ayarlayabilir ya da planlama amacıyla kullanım istatistiklerini toplayabilirsiniz.

Bir API çıkışı çalıştırıldığında ne olur?

Bir çıkış programı yazdıktan ve bunu WebSphere MQolarak tanımlandıktan sonra, kuyruk yöneticisi çıkış kodunuzu otomatik olarak kayıtlı noktalarda çağırır.

Çalıştırılacak API çıkış yordamları IBM i, Windows, UNIX and Linux sistemlerinde stanzas olarak tanımlanır. This topic covers the stanzas in the configuration files mqs.ini and qm.ini.

Yordamların tanımı üç yerde oluşabilir:

1. ApiExitCommon, mqs.ini dosyasında, kuyruk yöneticileri başlatıldığında uygulanan WebSphere MQ' nun tamamı için yordamları tanımlar. Bunlar, tek tek kuyruk yöneticileri için tanımlanan yordamlar tarafından geçersiz kılınabilir (bu listedeki “3” sayfa 371 öğesine bakın).
2. ApiExitTemplate, in the mqs.ini file, identifies routines, for the whole of WebSphere MQ, copied to the ApiExitLocal set (see item “3” sayfa 371 in this list) when a new queue manager is created.
3. ApiExitYerel olarak, qm.ini dosyasında, belirli bir kuyruk yöneticisi için geçerli olan yordamları tanımlar.

When a new queue manager is created, the ApiExitTemplate definitions in mqs.ini are copied to the ApiExitLocal definitions in qm.ini for the new queue manager. Bir kuyruk yöneticisi başlatıldığında, hem ApiExitOrtak, hem de ApiExitYerel tanımlamaları kullanılır. The ApiExitLocal definitions replace the ApiExitCommon definitions if both identify a routine of the same name. “API çıkışlarını yapılandırma” sayfa 376 içinde açıklanan Sequence özniteliği, stanzas çalıştırmasında tanımlanan yordamların sıralarını belirler.

Birden çok WebSphere MQkuruluşu arasında API çıkışlarının kullanılması

7.1 sürümündeki çıkışlar için yapılan değişikliklerin önceki bir sürümle çalışmaması nedeniyle, WebSphere MQ ' un önceki sürümü için yazılan API çıkışlarının tüm sürümlerle çalışmak için kullanıldığından emin olun. Çıkışlar için yapılan değişikliklerle ilgili daha fazla bilgi için bkz. “Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357.

API, amqsaem ve amqsaxe için sağlanan örnekler, çıkışlar yazılırken gerekli değişiklikleri yansıtır. İstemci uygulaması, uygulamanın ilişkili olduğu kuyruk yöneticisinin kuruluşuna karşılık gelen doğru WebSphere MQ kitaplıklarının, uygulamanın başlatılmasından önce bu kitaplıklara bağlandığını doğrulamalıdır.

API çıkışları yazılıyor

C programlama dilini kullanarak her API çağrısı için çıkış yazabilirsiniz.

Her API çağrısı için aşağıdaki gibi tüm çıkışlar kullanılabilir:

- MQCB, belirtilen nesne tanıtıcısı ve denetim etkinleştirmesi için geri çağrıyı yeniden kaydettirmek ve geri bildirme için yapılan değişiklikleri denetlemek için
- MQCTL, bağlantı için açılan nesne tanıtıcılarında denetleme işlemlerini gerçekleştirmek için
- MQCONN/MQCONNX, sonraki API çağrılarında kullanılmak üzere kuyruk yöneticisi bağlantı tanıtıcısı sağlamak için
- MQDISC, kuyruk yöneticisinden bağlantıyı kesmek için
- MQBEGIN, genel iş birimini başlatmak için (UOW)
- MQBACK, bir UOW ' u yedeklemek için
- MQCMIT, bir UOW ' u kesinleştirmek için
- Sonraki erişim için bir WebSphere MQ kaynağını açmak için MQOPEN
- Daha önce erişim için açılmış olan bir WebSphere MQ kaynağını kapatmak için MQCLOSE
- Erişim için önceden açılmış bir kuyruktan ileti almak için MQGET
- MQPUT1, bir iletiyi kuyruğa yerleştirmek için
- MQPUT, daha önce erişim için açılmış olan bir kuyruğa ileti yerleştirecek
- MQINQ, daha önce erişim için açılmış olan bir WebSphere MQ kaynağının özniteliklerine ilişkin bilgi edinmek için
- Erişim için önceden açılmış bir kuyruğun özniteliklerini ayarlamak için MQSET
- MQSTAT, durum bilgilerini almak için
- MQSUB, uygulama aboneliğini belirli bir konuya kaydettirmek için
- Bir abonelik isteği yapmak için MQSUBRQ

MQ_CALLBACK_EXIT, geri bildirme işleminden önce ve sonra gerçekleştirilmek üzere bir çıkış işlevi sağlar. Ek bilgi için Callback-MQ_CALLBACK_EXITbaşlıklı konuya bakın.

API çıkışlarında, aramalar genel formu alır:

```
MQ_call_EXIT (parameters, context, ApiCallParameters)
```

Burada *call* , MQ öneki olmayan MQI adını belirtir; örneğin, PUT, GET. *parameters* , çıkış ve dış denetim blokları MQAXP (API çıkış değiştirgesi yapısı) ve MQAXC (API çıkış bağlamı yapısı)arasında iletişim sağlayan, çıkışa ilişkin işlevi denetler. *context* , API çıkışısının çağrıldığı bağlamı açıklar ve *ApiCallParameters* , MQI çağrısına ilişkin parametreleri gösterir.

API çıkışınızı yazmanıza yardımcı olması için örnek bir çıkış (amqsaxe0.c) sağlanır; bu çıkış, izleme girişlerini belirlediğiniz bir dosyaya oluşturur. Bu örneği, çıkışlar yazılırken başlangıç noktanız olarak kullanabilirsiniz. Örnek çıkığı kullanma hakkında daha fazla bilgi için bkz. "API çıkış örnek programı" sayfa 108.

API çıkış çağrıları, dış denetim blokları ve ilişkili konular hakkında daha fazla bilgi için bkz. API çıkış başvurusu.

Bir çıkışa nasıl yazılacağı, derleneceği ve yapılandırılmasına ilişkin genel bilgiler için bkz. "Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi" sayfa 357.

API çıkışlarında ileti tanıtıcıları kullanılıyor

Bir API çıkışısının erişimi olan ileti özelliklerini denetleyebilirsiniz. Özellikler, bir ExitMsgişleciyle ilişkilendirilir. Put exit (put) çıkışta ayarlanan özellikler, yerleştirilecek iletiye ayarlanır, ancak alma çıkışta alınan özellikler uygulamaya geri döndürülmez.

Function ile MQXF_INIT ve **ExitReason** MQXR_CONNECTION değerine ayarlanmış MQXEP MQI çağrısını kullanarak bir MQ_INIT_EXIT çıkış işlevini kaydettirdiğinizde, bir MQXEPO yapısını **ExitOpts** değiştirgesi olarak geçirmenizi sağlar. MQXEPO yapısı, çıkışta kullanılabilecek özellikler kümesini belirten ExitProperties (ExitProperties) alanını içerir. Özelliklerin önekini gösteren, bir MQRFH2 klasör adına karşılık gelen bir karakter dizgisi olarak belirtilir.

Her API çıkışı, bir ExitMsgHandle alanı içeren bir MQAXP yapısı alır. Bu alan, WebSphere MQ tarafından oluşturulan ve bir bağlantıya özgü bir değere ayarlıdır. Bu nedenle, aynı bağlantıda aynı ya da farklı tiplerde API çıkışları arasında değişmeden kalır.

In an MQ_PUT_EXIT or MQ_PUT1_EXIT with an **ExitReason** of MQXR_BEFORE, that is, an API exit performed before putting a message, any properties (other than message descriptor properties) associated with the ExitMsgHandle when the exit completes are set on the message being put. Bu işlemi önlemek için, ExitMsgtanıtıcısını MQHM_NONE olarak ayarlayın. Ayrıca, farklı bir ileti tanıtıcısı da sağlayabilirsiniz.

MQ_GET_EXIT ' de, ExitMsgHandle değeri, özellikler temizlenir ve MQ_INIT_EXIT, ileti tanımlayıcı özellikleri dışında, ExitProperties alanında belirtilen özelliklerle doldurulur. Bu özellikler, alma uygulaması tarafından kullanılabilir kılınmaz. Alma uygulaması, MQGMO (İleti seçenekleri al) alanında bir ileti tanıtıcısı belirlediyse, ileti tanımlayıcı özellikleri de içinde olmak üzere, o tanıtıcı ile ilişkilendirilmiş tüm özellikler API çıkışa kullanılabilir. ExitMsgHandle 'ın özelliklerle doldurulmasını önlemek için, bu değeri MQHM_NONE olarak ayarlayın.

API çıkışlarında ileti tutamaçlarının kullanımını göstermek için bir örnek program (amqsaem0.c) sağlanır.

API çıkışları derleniyor

Bir çıkış yazdıktan sonra, aşağıdaki şekilde derleyip bağlardınız.

Aşağıdaki örneklerde, "API çıkış örnek programı" sayfa 108içinde açıklanan örnek program için kullanılan komutlar gösterilmektedir. For platforms other than Pencereler systems, you can find the sample API exit code in *MQ_INSTALLATION_PATH*/samp and the compiled and linked shared library in *MQ_INSTALLATION_PATH*/samp/bin. Windows sistemleri için, örnek API çıkış kodunu *MQ_INSTALLATION_PATH*\Tools\c\Samples' ta bulabilirsiniz. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizini gösterir.

Kullanıcılara not:

1. 64 bit kullanan programlamaya ilişkin yönergeler, 64 bit altyapılarda Coding standartlarıiçinde listelenir.

Bazı iletiler kuyruk yöneticisinden geçemeyebileceğinden, çok hedefli istemciler, API çıkışlar ve veri dönüştürme çıkışlarının istemci tarafında çalışabilmesi için bu çıkışlar istemcide çalışır. Aşağıdaki kitaplıklar, sunucu paketlerinin yanı sıra, istemci paketlerinin bir parçası olarak da yer alıyor:

Çizelge 54. İstemci ve sunucu paketlerinde bulunan kitaplıklar

İşletim sistemi	**Kitaplıklar**
Pencereler	32 bit & 64 bit: mqm.dll & mqm.pdb
Linux & HP-UX	32 bit ve 64 bit: libmqm.so & libmqm_r.so
AIX	32 bit & 64 bit: libmqm.a & libmqm_r.a
Solaris	32 bit ve 64 bit: libmqm.so

Unix ve Linux sistemlerinde API çıkışlarının derlenmesi
UNIX ve Linux sistemlerinde API çıkışlarının nasıl derletebileceği örnekler.

Tüm altyapılarda, modüle giriş noktası MQStart 'tır.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

AIXüzerinde

API çıkışı kaynak kodunu derleyerek aşağıdaki komutlardan birini çalıştırın:

32 bit uygulamalar

İş parçacıklı olmayan

```
cc -e MQStart -bE:amqsaxe.exp -bM:SRE -o /var/mqm/exits/amqsaxe \
    amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
xlc_r -e MQStart -bE:amqsaxe.exp -bM:SRE -o /var/mqm/exits/amqsaxe_r \
    amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

64 bit uygulamalar

İş parçacıklı olmayan

```
cc -q64 -e MQStart -bE:amqsaxe.exp -bM:SRE -o /var/mqm/exits64/amqsaxe \
    amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
xlc_r -q64 -e MQStart -bE:amqsaxe.exp -bM:SRE -o /var/mqm/exits64/amqsaxe_r \
    amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

HP-UX Itanium platformunda

32 bit uygulamalar

İş parçacıklı olmayan

API çıkış kaynak kodunu derleyin:

```
c89 +e +z -c -D_HPUX_SOURCE -o amqsaxe.o amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

API Çıkış kaynak kodunu bağla

```
ld +b: -b amqsaxe.o +ee MQStart -o /var/mqm/exits/amqsaxe
rm amqsaxe.o
```

İş parçacıklı

API çıkış kaynak kodunu derleyin:

```
c89 +e +z -c -D_HPUX_SOURCE -o amqsaxe.o amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

API Çıkış kaynak kodunu bağla

```
ld +b: -b amqsaxe.o +ee MQStart -o /var/mqm/exits/amqsaxe_r
rm amqsaxe.o
```

64 bit uygulamalar

İş parçacıklı olmayan

API çıkış kaynak kodunu derleyin:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o amqsaxe.o amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

API Çıkış kaynak kodunu bağla

```
ld -b amqsaxe.o +ee MQStart -o  /var/mqm/exits64/amqsaxe
rm amqsaxe.o
```

İş parçacıklı

API çıkış kaynak kodunu derleyin:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o amqsaxe.o amqsaxe0.c -IMQ_INSTALLATION_PATH/inc
```

API Çıkış kaynak kodunu bağla

```
ld -b amqsaxe.o +ee MQStart -o  /var/mqm/exits64/amqsaxe_r
rm amqsaxe.o
```

AçıkLinux

API çıkışı kaynak kodunu derleyerek aşağıdaki komutlardan birini çalıştırın:

31 bit uygulamalar

İş parçacıklı olmayan

```
 gcc -m31 -shared -fPIC -o /var/mqm/exits/amqsaxe amqsaxe0.c \
     -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
gcc -m31 -shared -fPIC -o /var/mqm/exits/amqsaxe_r amqsaxe0.c \
    -IMQ_INSTALLATION_PATH/inc
```

32 bit uygulamalar

İş parçacıklı olmayan

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/amqsaxe amqsaxe0.c \
     -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/amqsaxe_r amqsaxe0.c \
    -IMQ_INSTALLATION_PATH/inc
```

64 bit uygulamalar

İş parçacıklı olmayan

```
gcc -m64 -shared -fPIC -o /var/mqm/exits64/amqsaxe amqsaxe0.c \
    -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
gcc -m64 -shared -fPIC -o /var/mqm/exits64/amqsaxe_r amqsaxe0.c \
    -IMQ_INSTALLATION_PATH/inc
```

Solaris üzerinde

API çıkışı kaynak kodunu derleyerek aşağıdaki komutlardan birini çalıştırın:

32 bit uygulamalar

SPARC altyapısı

```
cc -xarch=v8plus -KPIC -mt -G -o /var/mqm/exits/amqsaxe \
   amqsaxe0.c -IMQ_INSTALLATION_PATH/inc \
   -R/usr/lib/32 -lsocket -lnsl -ldl
```

x86-64 platformu

```
cc -xarch=386 -KPIC -mt -G -o /var/mqm/exits/amqsaxe \
   amqsaxe0.c -IMQ_INSTALLATION_PATH/inc \
   -R/usr/lib/32 -lsocket -lnsl -ldl
```

64 bit uygulamalar

SPARC altyapısı

```
cc -xarch=v9 -KPIC -mt -G -o /var/mqm/exits64/amqsaxe \
   amqsaxe0.c -IMQ_INSTALLATION_PATH/inc \
   -R/usr/lib/64 -lsocket -lnsl -ldl
```

x86-64 platformu

```
cc -xarch=amd64 -KPIC -mt -G -o /var/mqm/exits64/amqsaxe \
   amqsaxe0.c -IMQ_INSTALLATION_PATH/inc \
   -R/usr/lib/64 -lsocket -lnsl -ldl
```

Windows sistemlerinde

Örnek API çıkış programını (`amqsaxe0.c`, Windows) derleyin ve bağlayın.

Bildirge (manifest) dosyası, derlenmiş bir uygulamaya ya da DLL ' ye gömülebilen, sürümü içeren isteğe bağlı bir XML belgesidir.

Bu tür bir belgeniz yoksa, **mt** komutundaki -bildirge *manifest.file* parametresini kaldırın.

Adapt the commands in the examples in Şekil 75 sayfa 376 or Şekil 76 sayfa 376 to compile and link `amqsaxe0.c` on Pencereler . Komutlar, Microsoft Visual Studio 2005, 2008 ya da 2010 ile çalışır. Örnekler, WebSphere MQ `C:\Program Files\IBM\WebSphere MQ\tools\c\samples` dizininin yürürlükteki dizin olduğunu varsayar.

32 bit

```
cl /c /nologo /MD /Foamqsaxe0.obj amqsaxe0.c

link /nologo /dll /def:amqsaxe.def

amqsaxe0.obj \
     /manifest /out:amqsaxe.dll

mt -nologo -manifest amqsaxe.dll.manifest \
   -outputresource:amqsaxe.dll;2
```

Şekil 75. 32 bit Windowsüzerinde amqsaxe0.c derleme ve bağlantı

64 bit

```
cl /c /nologo /MD /Foamqsaxe0.obj amqsaxe0.c

link /nologo /dll /def:amqsaxe.def \
     /libpath:..\..\lib64 \


amqsaxe0.obj /manifest /out:amqsaxe.dll

mt -nologo -manifest amqsaxe.dll.manifest \
   -outputresource:amqsaxe.dll;2
```

Şekil 76. 64 bit Pencerelerüzerinde amqsaxe0.c derleme ve bağlantı

İlgili kavramlar

“API çıkış örnek programı” sayfa 108

Örnek API çıkışı, kullanıcı tarafından belirtilen bir dosyaya MQAPI_TRACE_LOGFILE ortam değişkeninde tanımlı bir önek içeren bir MQI izleme oluşturur.

API çıkışlarını yapılandırma

Yapılandırma bilgilerini değiştirerek API çıkışlarını etkinleştirmek için IBM WebSphere MQ özelliğini yapılandırıyorsunuz.

Yapılandırma bilgilerini değiştirmek için, çıkış yordamlarını tanımlayan stanzaları ve bunların çalıştırıldığı sırayı değiştirmelisiniz. Bu bilgiler aşağıdaki şekillerde değiştirilebilir:

- IBM WebSphere MQ Explorer (Açık Windows ve Linux (x86 ve x86-64 platformlarında))
- **amqmdain** komutunun kullanılması (Windowsüzerinde)
- mqs.ini ve qm.ini dosyalarının doğrudan kullanılması (Windows, UNIX and Linux sistemleri üzerinde).

  mqs.ini dosyası, belirli bir düğümdeki tüm kuyruk yöneticilerine ilişkin bilgileri içerir. You can find it in the `/var/mqm` directory on UNIX and Linux and in the WorkPath specified in the `HKLM\SOFTWARE\IBM\WebSphere MQ` key on Windows systems.

  qm.ini dosyası, belirli bir kuyruk yöneticisine ilişkin bilgileri içerir. Kuyruk yöneticisi tarafından meşgul edilen dizin ağacının kökünde tutulan her kuyruk yöneticisi için bir kuyruk yöneticisi yapılanış kütüğü vardır. Örneğin, QMNAME adı verilen bir kuyruk yöneticisine ilişkin bir yapılanış kütüğünün yolu ve adı:

  UNIX and Linux sistemlerinde:

  ```
  /var/mqm/qmgrs/QMNAME/qm.ini
  ```

  Windows sistemlerinde:

  ```
  C:\Program Files\IBM\WebSphere MQ\qmgrs\QMNAME\qm.ini
  ```

Bir yapılandırma dosyasını düzenlemeden önce, bir kopyaya sahip olmak için gereksinim duyarsa geri dönebileceğiniz bir dosyayı yedeklemeniz gerekir.

Yapılandırma dosyalarını da düzenleyebilirsiniz:

– Düğümdeki kuyruk yöneticilerinin yapılandırmasını değiştiren komutları otomatik olarak kullanma
– Standart bir metin düzenleyiciyi kullanarak el ile

Bir yapılandırma dosyası özniteliğe yanlış bir değer ayarladıysanız, değer yoksayılır ve sorunu belirtmek için bir işletmen iletisi yayınlanır. (Etki, özniteliği tamamen eksik olarak görmektedir.)

Yapılandırılacak stanzas

Değiştirilmesi gereken stanzalar şunlardır:

ApiExitOrtak

Defined in mqs.ini and in the IBM WebSphere MQ Explorer on the IBM WebSphere MQ properties page, under Exits.

Kuyruk yöneticisi başlatıldığında, bu stanza içindeki öznitelikler okunur ve qm.iniiçinde tanımlanan API çıkışlarıyla geçersiz kılınır.

ApiExitŞablonu

Defined in mqs.ini and in the IBM WebSphere MQ Explorer on the IBM WebSphere MQ properties page, under Exits.

Herhangi bir kuyruk yöneticisi yaratıldığında, bu stanza içindeki öznitelikler, ApiExitLocal stanza altındaki yeni oluşturulan qm.ini dosyasına kopyalanır.

ApiExitYerel

Defined in qm.ini and in the IBM WebSphere MQ Explorer on the queue manager properties page, under Exits.

Kuyruk yöneticisi başlatıldığında, burada tanımlanan API çıkışları mqs.iniiçinde tanımlanan varsayılan değerleri geçersiz kılar.

Stanzalara ilişkin öznitelikler

- API çıkışa aşağıdaki özniteliği kullanarak ad girin:

  Ad=ApiExit_name
  MQAXP yapısının ExitInfoAd alanında geçirilen API çıkışa ilişkin açıklayıcı ad.

  Bu ad benzersiz olmalı, 48 karakterden uzun olmamalıdır ve yalnızca IBM WebSphere MQ nesnelerinin adları (örneğin, kuyruk adları) için geçerli karakterler içermelidir.

- Aşağıdaki öznitelikleri kullanarak çalıştırmak için API çıkış kodunun modül ve giriş noktasını tanımlayın:

  Function=function_name
  İşlev giriş noktasının adı, API çıkış kodunu içeren modüle işaret eder. Bu giriş noktası, MQ_INIT_EXIT işlevidir.

  Bu alanın uzunluğu, MQ_EXIT_NAME_LENGTH ile sınırlıdır.

  Module=modüle_adı
  API çıkış kodunu içeren modül.

  Bu alan, olduğu gibi kullanılan modülün tam yol adını içeriyorsa.

  Bu alan yalnızca modül adı içeriyorsa, modül qm.iniiçindeki `ExitPath` içindeki `ExitsDefaultPath` özniteliği kullanılarak bulunur.

  Ayrı iş parçacıklı kitaplıkları destekleyen altyapılarda, API çıkış modülünün iş parçacıklı ve iş parçacıklı bir sürümünü sağlamanız gerekir. Yivli sürümdeki bir `_r` soneki olmalıdır. IBM WebSphere

MQ uygulama sınırlı kod öbeğinin iş parçacıklı sürümü, yüklenmeden önce belirtilen modül adına örtük olarak `_r` ' yi ekler.

Bu alanın uzunluğu, platformun desteklediği yol uzunluğu üst sınırı ile sınırlıdır.

- İsteğe bağlı olarak, aşağıdaki özniteliği kullanarak çıkışa veri iletin:

  Veri=veri_adı
  MQAXP yapısındaki ExitData alanında API çıkışa geçirilecek veriler.

  Bu özniteliği eklerseniz, baştaki ve sondaki boşluklar kaldırılırsa, kalan dizgi 32 karaktere kesilir ve sonuç çıkışa geçirilir. Bu özniteliği atlarsanız, çıkışa 32 boşluktan oluşan varsayılan değer iletilir.

  Bu alanın uzunluk üst sınırı 32 karakterdir.

- Aşağıdaki özniteliği kullanarak diğer çıkışlarla ilgili olarak bu çıkışa ilişkin sırayı tanımlayın:

  Sequence=sıra_numarası
  Bu API çıkışısının diğer API çıkışlarına göre çağrıldığı sıra. Sıra numarası düşük olan bir çıkış, daha yüksek sıra numarasına sahip bir çıkıştan önce çağrılır. Çıkışların sıra numaralandırmasına bitişik olacak şekilde gerek yoktur. 1, 2, 3 gibi bir sıra, 7, 42, 1096 diziyle aynı sonucu elde eder. İki çıkış aynı sıra numarasına sahip olursa, kuyruk yöneticisi hangisinin önce arayacağına karar verir. MQAXP ' de ExitChainAreaPtr tarafından belirtilen ExitChainArea alanına saati ya da bir imleyiciyi koyarak ya da kendi günlük dosyanızı yazarak, olayın ardından hangilerinin çağrıldığını söyleyebilirsiniz.

  Bu öznitelik işaretsiz bir sayısal değer.

Örnek stanzas

Örnek mqs.ini dosyası aşağıdaki stanzaları içerir:

ApiExitŞablonu
Bu stanza, açıklayıcı adı `OurPayrollQueueAuditor`, modül adı `auditor`ve sıra numarası 2 ile bir çıkış tanımlar. Çıkışa 123 veri değeri iletilir.

ApiExitOrtak
This stanza defines an exit with the descriptive name `MQPoliceman`, module name `tmqp`, and sequence number 1. İletilen veriler bir yönergedir (`CheckEverything`).

```
mqs.ini

   ApiExitTemplate:
      Name=OurPayrollQueueAuditor
      Sequence=2
      Function=EntryPoint
      Module=/usr/ABC/auditor
      Data=123
   ApiExitCommon:
      Name=MQPoliceman
      Sequence=1
      Function=EntryPoint
      Module=/usr/MQPolice/tmqp
      Data=CheckEverything
```

Aşağıdaki örnek qm.ini dosyası, tanımlayıcı adı `ClientApplicationAPIchecker`, birim adı `ClientAppChecker`ve sıra numarası 3 ile çıkışa ilişkin bir ApiExitYerel tanımlaması içerir.

```
qm.ini

   ApiExitLocal:
      Name=ClientApplicationAPIchecker
      Sequence=3
      Function=EntryPoint
      Module=/usr/Dev/ClientAppChecker
      Data=9.20.176.20
```

İleti alışverişi kanallarına ilişkin kanal çıkışı programları

Bu konu derlemi, ileti alışverişi kanallarına ilişkin WebSphere MQ kanal çıkış programlarıyla ilgili bilgileri içerir.

İleti kanalı aracıları (MCA ' lar) veri dönüştürme çıkışları da çağırabilir. Veri dönüştürme çıkışları yazma hakkında daha fazla bilgi için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397.

Bu bilgilerin bazıları, WebSphere MQ MQI istemcilerini kuyruk yöneticilerine bağlayan MQI kanallarındaki çıkışlar için de geçerlidir. Ek bilgi için MQI kanallarına ilişkin kanal çıkışı programlarıbaşlıklı konuya bakın.

Kanal çıkış programları, MCA programları tarafından gerçekleştirilen işlemde tanımlı yerlerde çağrılır.

Bu kullanıcı çıkışı programlarından bazıları tamamlayıcı çiftlerde çalışır. Örneğin, ileti gönderme işlevi tarafından iletilecek iletileri şifrelemek için bir kullanıcı çıkışı programı çağrılırsa, süreci tersine çevirmek için tamamlayıcı işlem, alma uçta çalışır durumda olmalıdır.

Çizelge 55 sayfa 379 , her kanal tipi için kullanılabilir olan kanal çıkışı tiplerini gösterir.

Çizelge 55. Her kanal tipi için kanal çıkışları kullanılabilir

Kanal Tipi	İleti çıkışı	İleti-çıkışı yeniden dene	Çıkış al	Güvenlik Çıkışı	Çıkış gönder	Otomatik tanımlama çıkışı
Gönderen kanalı	Evet		Evet	Evet	Evet	
Sunucu kanalı	Evet		Evet	Evet	Evet	
Küme-gönderen kanalı	Evet		Evet	Evet	Evet	Evet
Alıcı kanalı	Evet	Evet	Evet	Evet	Evet	Evet
İstekte bulunanın kanalı	Evet	Evet	Evet	Evet	Evet	
Küme-alıcı kanalı	Evet	Evet	Evet	Evet	Evet	Evet
İstemci bağlantı kanalı			Evet	Evet	Evet	
Sunucu bağlantısı kanalı			Evet	Evet	Evet	Evet

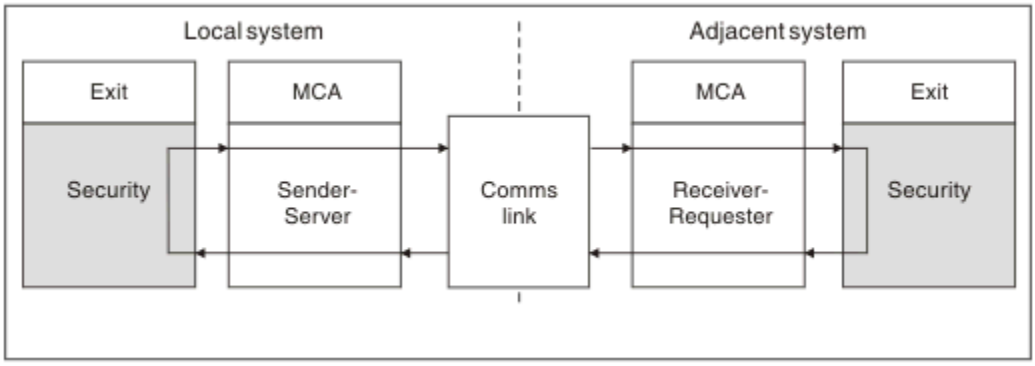
Bir istemcide kanal çıkışlarını çalıştırabildiyseniz, MQSERVER ortam değişkenini kullanamazsınız. Bunun yerine, İstemci kanal tanımlama çizelgesi içinde açıklandığı gibi bir istemci kanal tanımlama çizelgesi (CCDT) yaratın ve başvurulayın.

İşleme genel bakış

MCA ' ların kanal çıkış programlarını nasıl kullanlarına genel bakış.

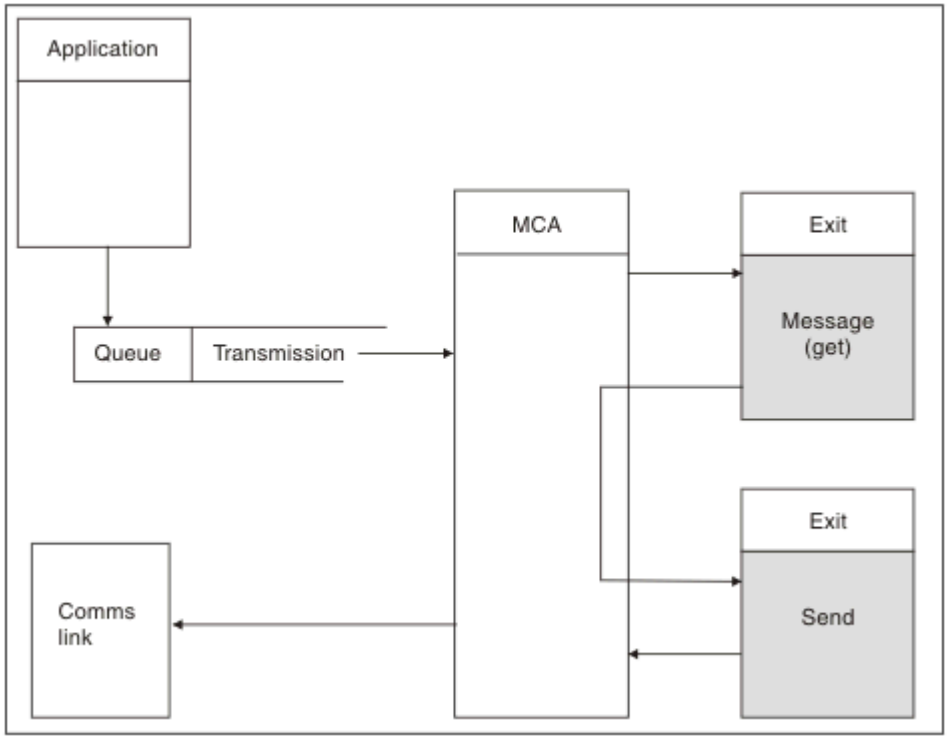
Başlatma sırasında, MCA ' lar işlemeyi eşitlemek için bir başlatma iletişim kutusu değiştirir. Daha sonra, güvenlik çıkışlarını içeren bir veri değiş tokasına geçiyorlar. Başlatma aşamasını tamamlamak ve iletilerin aktarılmasına izin vermek için bu çıkışlar başarıyla sona ermelidir.

Güvenlik denetimi aşaması, Şekil 77 sayfa 380içinde gösterildiği gibi bir döngüdür.

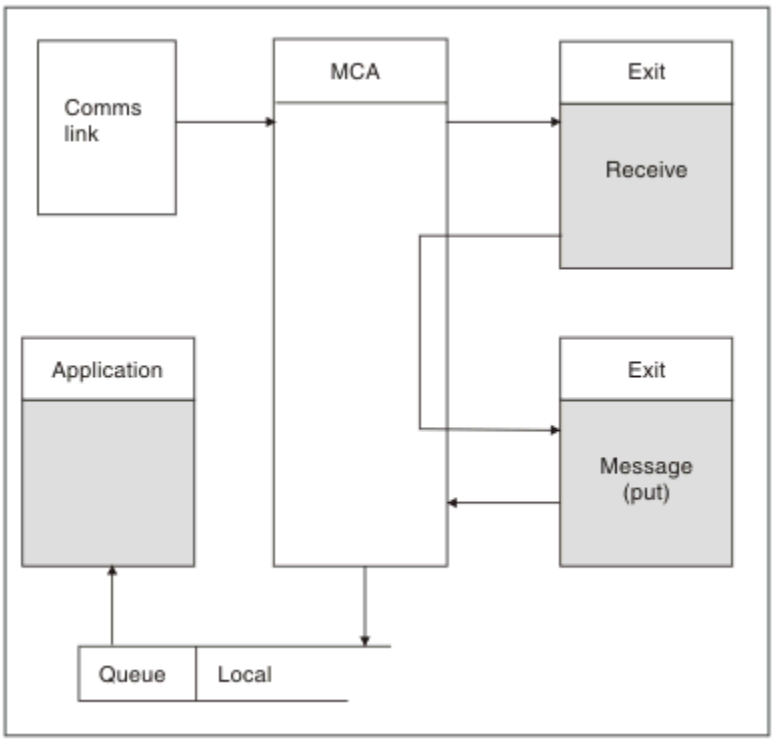


Şekil 77. Güvenlik çıkış döngüsü

İleti aktarma aşaması sırasında, MCA 'yı gönderme işlemi iletileri bir iletim kuyruğundan alır, ileti çıkışını çağırır, gönderme çıkışını çağırır ve Şekil 78 sayfa 380içinde gösterildiği şekilde iletiyi alma MCA' sına gönderir.



Şekil 78. İleti kanalının gönderici bitişindeki gönderme çıkışı örneği



Şekil 79. İleti kanalının günlük nesnesindeki alma çıkışa örneği

The receiving MCA receives a message from the communications link, calls the receive exit, calls the message exit, and then puts the message on the local queue, as shown in Şekil 79 sayfa 381. (Alma çıkışa, ileti çıkışı çağrılmadan önce bir kereden fazla çağrılabilir.)

Kanal-çıkış programları yazılıyor

Kanal çıkış programları yazmanıza yardımcı olması için aşağıdaki bilgileri kullanabilirsiniz.

Kullanıcı çıkışları ve kanal çıkışı programları, izleyen kısımlarda belirtilenler dışında tüm MQI çağrılarını kullanabilir. MQ V7 ve sonraki sürümleri için, MQCXP yapısı sürüm 7 ve üstü, MQCONN komutu vermek yerine kullanılabilecek hConnbağlantı tanıtıcısını içerir. Önceki sürümler için, kanal kendisi kuyruk yöneticisine bağlı olduğundan, bir MQRC_ALREADY_CONNECTED uyarısı döndürülse de, bağlantı tanıtıcısını almak için bir MQCONN yayınlanmalıdır.

Kanal çıkışısının iş parçacığı korumalı olması gerektiğini unutmayın.

İstemci-bağlantı kanallarındaki çıkışlar için, çıkışa bağlanmayı denediği kuyruk yöneticisi, çıkışa nasıl bağlı olduğuna bağlıdır. Çıkış MQM.LIB ile bağlantılıysa ve MQCONN çağrısında bir kuyruk yöneticisi adı belirtmezseniz, çıkış, sisteminizdeki varsayılan kuyruk yöneticisine bağlanmayı dener. Çıkış MQM.LIB ile bağlantılıysa ve MQCD ' nin QMgrName alanı aracılığıyla çıkışa aktarılan kuyruk yöneticisinin adını belirtirseniz, çıkış o kuyruk yöneticisine bağlanmayı dener. Çıkış MQIC.LIB ya da başka bir kitaplık için, MQCONN çağrısı bir kuyruk yöneticisi adı belirtme ya da not belirtme belirtmediğiniz için başarısız olur.

You should avoid altering the state of the transaction associated with the passed hConn in a channel exit; you must not use the MQCMIT, MQBACK or MQDISC verbs with the channel hConn, and you cannot use the MQBEGIN verb specifying the channel hConn.

MQCONNX, yeni bir IBM WebSphere MQ bağlantısı yaratmak için MQCNO_HANDLE_SHARE_BLOCK ya da MQCNO_HANDLE_SHARE_NO_BLOCK belirtilerek kullanılıyorsa, bağlantının doğru olarak yönetilmesini ve kuyruk yöneticisinden bağlantıyı doğru olarak kesmesini sağlamak sizin sorumluluğunuzda olur. Örneğin, bağlantı kesmeden her çağrıda kuyruk yöneticisine yeni bir bağlantı oluşturan bir kanal çıkışı, bağlantı tanıtıcılarının oluşturulması ve aracı iş parçacıklarının sayısının artmasını sağlar.

Bir çıkış MCA ' nın kendisi ile aynı iş parçacığıda çalışır ve aynı bağlantı tanıtıcısını kullanır. Bu nedenle, MCA ile aynı UOW ' un içinde çalışır ve tutarlılık noktası altında yapılan tüm aramalar, toplu işin sonundaki kanaldan kesinleştirilir ya da yedeklenir.

Bu nedenle, bir kanal ileti çıkışı, özgün iletiyi içeren toplu iş kesinleştirildiğinde, yalnızca o kuyruk için kesinleştirilen bildirim iletileri gönderebilir. Bu nedenle, bir kanal iletisi çıkışından Sync Point MQI çağrıları yayınlanıyor olabilir.

Bir kanal çıkışı, MQCD ' deki alanları değiştirebilir. Ancak, bu değişikliklerin listelendiği durumlar dışında, bu değişiklikler üzerinde işlem yapmamış olabilir. Bir kanal çıkış programı, MQCD veri yapısındaki bir alanı değiştirirse, yeni değer IBM WebSphere MQ kanal işlemi tarafından yok sayılır. Ancak, yeni değer MQCD ' de kalır ve bir çıkış zincirindeki geri kalan çıkışlara ve kanal yönetim ortamını paylaşan herhangi bir konuşmaya geçirilir. Ek bilgi için Kanal çıkışta MQCD alanlarının değiştirilmesibaşlıklı konuya bakın.

Ayrıca, C kitaplığında yazılan programlar için, yeniden giriş-dışı C kitaplığı işlevi, kanal çıkış programında kullanılmamalıdır.

Aynı anda birden çok kanal çıkış kitaplığı kullanıyorsanız, iki farklı çıkışa ilişkin kod aynı şekilde adlandırılmış işlevler içeriyorsa, bazı UNIX and Linux platformlarında sorunlar ortaya çıkabilir. Bir kanal çıkışı yüklendiğinde, dinamik yükleyici çıkış kitaplığındaki işlev adlarını kitaplığın yüklendiği adreslere çözer. İki çıkış kitaplığı, aynı adlara sahip ayrı işlevler tanımlarsa, bu çözüm süreci, bir kitaplığın işlev adlarını başka bir kitaplığın işlevlerini kullanabilecek şekilde çözebilir. Bu sorun ortaya çıkarsa, bu işlevler etkilenmeden yalnızca gerekli çıkışı ve MQStart işlevlerini dışa aktarması gerektiğini linker ' a belirtin. Diğer işlevlere, kendi çıkış kitaplıklarının dışındaki işlevler tarafından kullanılmamaları için yerel görünürlük verilmelidir. Ek bilgi için bağlantı oluşturucuya ilişkin belgelere bakın.

Tüm çıkışlar, bir kanal çıkış parametresi yapısı (MQCXP), bir kanal tanımlama yapısı (MQCD), hazırlanmış veri arabelleği, veri uzunluğu parametresi ve arabellek uzunluğu parametresiyle çağrılır. Arabellek uzunluğu aşılmamalı:

- İleti çıkışları için, kanal genelinde gönderilmesi gereken en büyük iletiye ve MQXQH yapısının uzunluğuna izin vermelisiniz.
- Gönderme ve alma çıkışları için, izin vermeniz gereken en büyük arabellek aşağıdaki gibidir:

  LU 6.2
  32 KB

  TCP:
  32 KB

  Not: Kullanılabilir uzunluk üst sınırı, bu uzunluğa göre 2 bayt daha az olabilir. Ayrıntılar için MaxSegmentUzunluğu altında döndürülen değeri denetleyin. MaxSegmentLength ile ilgili daha fazla bilgi için bkz. MaxSegmentUzunluğu.

  NetBIOS:
  64 KB

  SPX:
  64 KB

  Not: Alıcı kanallarındaki gönderici kanallardan ve gönderici çıkışlardan, TCP için 2 KB arabellekten çıkar çıkar.
- Güvenlik çıkışları için, dağıtılmış kuyruğa alma olanağı 4000 baytlık bir arabellek ayırır.

Çıkışta, ilgili değiştirgelerle birlikte alternatif bir arabelleğin döndürülmesi olanaklıdır. Arama ayrıntıları için “İleti alışverişi kanallarına ilişkin kanal çıkışı programları” sayfa 378 ' e bakın.

Writing channel-exit programs on Pencereler, UNIX and Linux systems
You can use the following information to help you write channel-exit programs for Pencereler, UNIX and Linux systems.

“Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357içinde belirtilen yönergeleri izleyin. Uygun durumlarda, aşağıdaki kanal çıkışından özel bilgileri kullanın:

Çıkış C ' de yazılmalı ve Windows 'ta bir DLL olmalıdır.

Define a dummy MQStart() routine in the exit and specify `MQStart` as the entry point in the library. Şekil 80 sayfa 383 , programınıza bir girdi nasıl ayarlayacağını gösterir:

```
#include <cmqec.h>

void MQStart() {;} /* dummy entry point - for consistency only */
void MQENTRY ChannelExit ( PMQCXP  pChannelExitParms,
                           PMQCD   pChannelDefinition,
                           PMQLONG pDataLength,
                           PMQLONG pAgentBufferLength,
                           PMQVOID pAgentBuffer,
                           PMQLONG pExitBufferLength,
                           PMQPTR  pExitBufferAddr)
{
... Insert code here
}
```

Şekil 80. Kanal çıkışa ilişkin örnek kaynak kodu

Visual C + + kullanan Windows için kanal çıkışları yazarken, kendi DEF dosyanızı yazmanız gerekir. Şekil 81 sayfa 383' ta nasıl gösterildiğini gösteren bir örnek. Kanal çıkışı programlarının yazılmasına ilişkin ek bilgi için “Kanal-çıkış programları yazılıyor” sayfa 381başlıklı konuya bakın.

```
EXPORTS
ChannelExit
```

Şekil 81. Pencereleriçin örnek DEF dosyası

Kanal güvenliği çıkış programları
Bir kanalın diğer ucundaki ortağın gerçek olduğunu doğrulamak için güvenlik çıkış programlarını kullanabilirsiniz. Bu, kimlik doğrulama olarak bilinir. Bir kanalın güvenlik çıkışı kullanması gerektiğini belirtmek için, kanal tanımının SCYEXIT alanında çıkış adını belirtin.

Not: Kimlik doğrulaması, kanal kimlik doğrulama kayıtlarıyla da gerçekleştirilebilir. Kanal doğrulama kayıtları , belirli kullanıcılardan ve kanallardan kuyruk yöneticilerine erişimin önlenmesinde ve uzak kullanıcıların IBM WebSphere MQ kullanıcı kimliklerine eşlenmesinde büyük esneklik sağlar. Kullanıcılarınızın kimliğini doğrulamak ve verileriniz için şifreleme ve veri bütünlüğü denetlemesi sağlamak için IBM WebSphere MQ tarafından SSL ve TLS desteği de sağlanır. SSL ve TLS hakkında daha fazla bilgi için bkz. WebSphere MQ support for SSL and TLS. Ancak, yine de güvenlik işlemleri için daha karmaşık (ya da farklı) formlara ve diğer denetim türlerine ve güvenlik bağlamına gerek duyuyorsanız, güvenlik çıkışlarını yazmayı düşünün.

For security exits written prior to IBM WebSphere MQ Version 7.1 it is worth noting that earlier versions of IBM WebSphere MQ queried the underlying secure sockets provider (e.g. GSKit) to determine the remote partner's certificate Subject Distinguished Name (SSLPEER) and Issuer Distinguished Name (SSLCERTI). In IBM WebSphere MQ Version 7.1 support was added for a range of new security attributes. Bu özniteliklere erişmek için IBM WebSphere MQ Version 7.1 , sertifikanın DER kodlamasını alır ve Konu ve Sertifika Veren DN ' ini belirlemek için bu kodlamayı kullanır. Aşağıdaki kanal durumu özniteliklerinde Konu ve Sertifika Veren DN öznitelikleri görüntülenir:

- SSLPEER (PCF seçici MQCACH_SSL_SHORT_PEER_NAME)
- SSLCERTI (PCF seçici MQCACH_SSL_CERT_ISSUER_NAME)

Bu değerler, kanal durumu komutlarının yanı sıra, aşağıda gösterildiği gibi, listelenen kanal güvenliği çıkışlarına aktarılan verilerin de döndürülmesini sağlar:

- MQCD SSLPeerNamePtr
- MQCXP SSLRemCertIssNamePtr

IBM WebSphere MQ Version 7.1'ta, bir SERIALNUMERT özniteliği, Subject DN' de de yer alır ve uzak ortağın sertifikasına ilişkin seri numarasını içerir. Ayrıca, bazı DN öznitelikleri önceki yayın düzeylerinden farklı bir sırada döndürülür. Sonuç olarak, SSLPEER ve SSLCERTI alanlarının bileşimi, önceki yayın düzeylerinden Version 7.1 ' de değiştirilir ve bu nedenle, bu alanlara bağımlı olan tüm güvenlik çıkışlarının ya da uygulamaların incelenip güncellenmemesine neden olur.

Bir kanal tanımlamasının SSLPEER alanı aracılığıyla belirtilen var olan WebSphere MQ eşdüzey ad süzgeçleri etkilenmez ve daha önceki yayınlarda olduğu gibi çalışmaya devam eder. This is because the WebSphere MQ peer name matching algorithm has been updated to process existing SSLPEER filters without any need to alter the channel definitions. Bu değişiklik büyük olasılıkla güvenlik çıkışlarını ve PCF programlama arabirimi tarafından döndürülen Sertifika Sahibi DN 'si ve Sertifika Veren DN değerlerine bağımlı olan uygulamaları etkiler.

Güvenlik çıkışı, C ya da Javaiçinde yazılabilir.

Kanal güvenlik çıkış programları, MCA ' nın işlem çevriminde aşağıdaki yerlerde çağrılır:

- MCA ' da kabul ve sonlandırma.
- Kanal başlatma sırasında ilk veri kararlaştığı sona erdikten hemen sonra. Kanalın alıcı ya da sunucu ucu, uzak uçta güvenlik çıkışa teslim edilmesi için bir ileti sağlayarak uzak ucuyla bir güvenlik iletisi değiş tokası başlatabilir. Ayrıca, bunu yapmayı da reddedebilir. Çıkış programı, uzak uçtan alınan her güvenlik iletisini işlemek için yeniden başlatılır.
- Kanal başlatma sırasında ilk veri kararlaştığı sona erdikten hemen sonra. Kanalın gönderen ya da istekte bulunan ucu, uzak uçtan alınan bir güvenlik iletisini işlerken ya da uzak sona erdirilemeyeceğini bildiren bir güvenlik değiş tokadı başlatır. Çıkış programı, alınabilecek sonraki tüm güvenlik iletilerini işlemek için yeniden başlatılır.

Bir istekte bulunanın kanalı hiçbir zaman MQXR_INIT_SEC ile çağrılamaz. Kanal, sunucuya bir güvenlik çıkış programı olduğunu bildirir ve sunucu, daha sonra bir güvenlik çıkışı başlatma olanağına sahiptir. Bir değer yoksa, istekte bulunana bildirir ve çıkış programına sıfır uzunluklu bir akış döndürülür.

Not: Sıfır uzunluklu güvenlik iletileri göndermekten kaçının.

Güvenlik çıkışı programlarıyla değiş tokuş edilen verilerin örnekleri Şekil 82 sayfa 385 ile Şekil 85 sayfa 387arasındaki şekillerle gösterilir. Bu örnekler, alıcısının güvenlik çıkışı ile gönderenin güvenlik çıkışıyla ilgili olayların sırasını gösterir. Rakamlardaki ardışık satırlar, zaman geçişini temsil eder. Bazı durumlarda, alıcı ve göndericindeki olaylar ilintili değildir ve bu nedenle aynı anda ya da farklı zamanlarda gerçekleşebilir. Diğer durumlarda, bir çıkış programındaki bir olay, diğer çıkış programında daha sonra oluşan tamamlayıcı bir olaya neden olur. Örneğin, Şekil 82 sayfa 385içinde:

1. Alıcı ve gönderen her biri MQXR_INIT ile çağrılır, ancak bu çağrılar ilintili değildir ve bu nedenle aynı anda ya da farklı zamanlarda ortaya çıkabilir.
2. Alıcı bir sonraki MQXR_INIT_SEC ile çağrılır, ancak gönderici çıkışında tamamlayıcı bir olay gerektirmeyecek MQXCC_OK değerini döndürür.
3. Gönderen bir sonraki MQXR_INIT_SEC ile çağrılır. Bu, alıcının MQXR_INIT_SEC ile çağrısıyla ilintili değildir. Gönderen, alıcı çıkışında tamamlayıcı bir olaya neden olan MQXCC_SEND_SEC_MSG değerini döndürür.
4. Daha sonra, alıcı MQXR_SEC_MSG ile çağrılır ve gönderici çıkışında tamamlayıcı bir olaya neden olan MQXCC_SEND_SEC_MSG döndürülür.
5. Daha sonra, gönderen MQXR_SEC_MSG ile çağrılır ve alıcı çıkışında tamamlayıcı bir olay gerektirmeyecek MQXCC_OK değerini döndürür.

Receiver exit	Sender exit
Invoked with MQXR_INIT Responds with MQXCC_OK	Invoked with MQXR_INIT Responds with MQXCC_OK
Invoked with MQXR_INIT_SEC Responds with MQXCC_OK	
	Invoked with MQXR_INIT_SEC Responds with MQXCC_SEND_SEC_MSG
Invoked with MQXR_SEC_MSG Responds with MQXCC_SEND_SEC_MSG	
	Invoked with MQXR_SEC_MSG Responds with MQXCC_OK
Message transfer begins	

Şekil 82. Sözleşme ile gönderici tarafından başlatılan değiş tokuş

Receiver exit	Sender exit
Invoked with MQXR_INIT Responds with MQXCC_OK	Invoked with MQXR_INIT Responds with MQXCC_OK
Invoked with MQXR_INIT_SEC Responds with MQXCC_OK	
	Invoked with MQXR_INIT_SEC Responds with MQXCC_SEND_SEC_MSG
Invoked with MQXR_SEC_MSG Responds with MQXCC_OK	
	Invoked with MQXR_SEC_MSG Responds with MQXCC_SUPPRESS_FUNCTION
Channel closes	
Invoked with MQXR_TERM Responds with MQXCC_OK	Invoked with MQXR_TERM Responds with MQXCC_OK

Şekil 83. Sözleşme olmadan gönderen ile başlatılan değiş tokuş

Receiver exit	Sender exit
Invoked with MQXR_INIT Responds with MQXCC_OK	Invoked with MQXR_INIT Responds with MQXCC_OK
Invoked with MQXR_INIT_SEC Responds with MQXCC_SEND_SEC_MSG	
	Invoked with MQXR_SEC_MSG Responds with MQXCC_SEND_SEC_MSG
Invoked with MQXR_SEC_MSG Responds with MQXCC_OK	
Message transfer begins	
Invoked with MQXR_TERM Responds with MQXCC_OK	Invoked with MQXR_TERM Responds with MQXCC_OK

Şekil 84. Alıcı ile sözleşme ile başlatılan değiş tokuş

Receiver exit	Sender exit
Invoked with MQXR_INIT Responds with MQXCC_OK	Invoked with MQXR_INIT Responds with MQXCC_OK
Invoked with MQXR_INIT_SEC Responds with MQXCC_SEND_SEC_MSG	
	Invoked with MQXR_SEC_MSG Responds with MQXCC_OK
Invoked with MQXR_SEC_MSG Responds with MQXCC_SUPPRESS_FUNCTION	
Channel closes	

Şekil 85. Sözleşme olmadan alıcı tarafından başlatılan değiş tokuş

Kanal güvenlik çıkış programı, güvenlik çıkışı tarafından oluşturulan iletim üstbilgileri dışında, güvenlik verilerini içeren bir aracı arabelleğinden geçirilir. Bu veriler, kanaldan her iki ucunun güvenlik doğrulaması gerçekleştirebilmesi için uygun bir veri olabilir.

İleti kanalının gönderme ve alma sonundaki güvenlik çıkış programı, herhangi bir çağrıya iki yanıt kodundan birini döndürebilir:

- Güvenlik değiş tokası hata olmadan sona erdi
- Kanalı engelle ve kapat

Not:

1. Kanal güvenlik çıkışları genellikle çiftler halinde çalışılır. Uygun kanalları tanımladığınızda, kanalın her iki ucu için uyumlu çıkış programlarının adlandırıldığından emin olun.
2. IBM i' ta, "Kabul edilmiş yetkiyi kullan" (USEADPAUT = *YES) ile derlenmiş güvenlik çıkış programları QMQM ya da QMQMADM yetkisini benimsibilir. Çıkışa dikkat edin, bu özelliği sisteminizin güvenlik riski taşıması için kullanmaz.
3. Kanalın diğer ucunun bir sertifika sağladığı bir SSL kanalında, güvenlik çıkışı, SSLPeerNamePtr tarafından erişilen MQCD alanına ve SSLRemCertIssNamePtr tarafından erişilen MQCXP alanına verenin Ayırt Edici Adı ile bu sertifikanın konularının ayırt edici adını alır. Bu adın koyabileceği kullanım alanları şunlardır:
   - Erişimi SSL kanalı üzerinden kısıtlamak için.
   - MQCD.MCAUserIdentifier adı temelinde.

İlgili kavramlar

Kanal doğrulama kayıtları

Güvenli Yuva Katmanı (SSL) ve İletim Arabirimi Katmanı Güvenliği (TLS) kavramları

Güvenlik çıkışı yazılıyor

güvenlik çıkış iskelet kodunu kullanarak bir güvenlik çıkışı yazabilirsiniz.

Şekil 86 sayfa 388 , bir güvenlik çıkışısının nasıl yazılacağını gösterir.

```
void MQENTRY MQStart() {;}
void MQENTRY EntryPoint (PMQVOID pChannelExitParms,
                         PMQVOID pChannelDefinition,
                         PMQLONG pDataLength,
                         PMQLONG pAgentBufferLength,
                         PMQVOID pAgentBuffer,
                         PMQLONG pExitBufferLength,
                         PMQPTR  pExitBufferAddr)
{
  PMQCXP pParms = (PMQCXP)pChannelExitParms;
  PMQCD  pChDef = (PMQCD)pChannelDefinition;
  /* TODO: Add Security Exit Code Here */

}
```

Şekil 86. Güvenlik çıkışı İskelet kodu

Standart WebSphere MQ Giriş Noktası MQStart var olmalıdır, ancak herhangi bir işlev gerçekleştirmek için gerekli değildir. İşleve ilişkin ad (bu örnekteEntryPoint) değiştirilebilir, ancak kitaplık derlenip bağlandığında işlev dışa aktarılmalıdır. Önceki örnekte olduğu gibi, işaretçiler pChannelExitParms , PMQCXP ve pChannelDefinition için PMQCD ' ye dönüştürülmelidir. Kanal çıkışlarının çağrılmasına ve parametrelerin kullanılmasına ilişkin genel bilgiler için MQ_CHANNEL_EXITbaşlıklı konuya bakın. Bu değiştirgeler bir güvenlik çıkışta aşağıdaki gibi kullanılır:

PMQVOID pChannelExitParms

giriş/çıkış

MQCXP yapısına ilişkin gösterge, alanlara erişmek için PMQCXP ' ye çevrilecek. Bu yapı, Çıkış ve MCA arasında iletişim kurmak için kullanılır. MQCXP ' deki aşağıdaki alanlar, Security Exits ile ilgili özel ilgi alanlarıdır:

ExitReason
Security Exit 'e güvenlik alışverişindeki mevcut durumu bildirir ve hangi işlemin yapılması kararlaştırılırken kullanılır.

ExitResponse
Güvenlik alışverişindeki bir sonraki aşamayı belirleyen MCA ' ya verilen yanıt.

ExitResponse2
MCA ' nın Güvenlik Çıkışı yanıtını nasıl yorumlayacağını yönetmek için ek denetim işaretleri.

ExitUserAlanı
Aramalar arasında durumu korumak için Security Exit (Güvenlik Çıkışı) tarafından kullanılabilecek 16 bayt (üst sınır).

ExitData
Kanal tanımlamasının SCYDATA alanında belirlenen verileri (boşluklarla sağa doldurulmuş 32 bayt) içerir.

PMQVOID pChannelTanımlaması
giriş/çıkış

Alanlara erişmek için MQCD yapısı-PMQCD ' ye dönüştürme yapın. Bu değiştirge, kanala ilişkin tanımlamayı içerir. MQCD ' deki aşağıdaki alanlar, Security Exits ile ilgili özel ilgi alanlarıdır:

ChannelName
Kanal adı (boşluklarla sağa doğru 20 bayt doldurulur).

ChannelType
Kanal tipini tanımlayan bir kod.

MCA Kullanıcı Kimliği
Bu üç alan grubu, kanal tanımında belirlenen MCAUSER alanı değeriyle ilk kullanıma hazırlandı. Bu alanlarda Güvenlik Çıkışı tarafından belirlenen herhangi bir kullanıcı kimliği erişim denetimi için kullanılır (SDR, SVR, CLNTCONN ya da CLUSSDR kanalları için geçerli değildir).

MCAUserIdentifier
İlk 12 baytlık tanıtıcı boşluklarla dolduruldu.

LongMCAUserIdPtr
Tam uzunluk tanıtıcısını içeren bir arabelleğe (garantili boş değer sonlandırılmamış) ilişkin gösterge, MCAUserIdentifier' un üzerinde önceliği alır.

LongMCAUserIdLength
Length of string pointed to by LongMCAUserIdPtr - must be set if LongMCAUserIdPtr is set.

Uzak Kullanıcı Tanıtıcısı
Yalnızca CLNTCONN/SVRCONN kanal çiftleri için geçerlidir. CLNTCONN Security Exit tanımlanmadıysa, bu üç alan istemci MCA tarafından başlatılır; böylece, kimlik doğrulama için bir SVRCONN Güvenlik Çıkışı ve MCA Kullanıcı Tanıtıcısı belirtilirken kullanılabilecek istemci ortamından bir kullanıcı kimliği içerebilirler. Bir CLNTCONN Security Exit tanımlıysa, bu alanlar kullanıma hazırlanmaz ve CLNNTCONN Security Exit ile ayarlanabilir ya da güvenlik iletileri, bir kullanıcı kimliğini İstemciden Sunucuya geçirmek için kullanılabilir.

RemoteUserTanıtıcısı
İlk 12 bayt tanıtıcısı boşluklarla dolduruldu.

LongRemoteUserIdPtr
Pointer to a buffer containing the full length identifier (not guaranteed null terminated) takes priority over RemoteUserIdentifier.

LongRemoteUserIdUzunluğu
LongRemoteUserIdPtr-byPtrPtr ayarlandıysa, LongRemoteUserIdPtr tarafından işaret edilen dizgi uzunluğu ayarlanmalıdır.

PMQUT pDataUzunluğu
giriş/çıkış

İşaretçiyi MQUZE. Security Exit (Güvenlik Çıkışı) çağrısının ardından AgentBuffer (AgentArabelleği) içinde bulunan herhangi bir Güvenlik Çıkışı uzunluğunu içerir. Must be set by a Security Exit to the length of any message being sent in the AgentBuffer or ExitBuffer.

PMQHOT pAgentBufferLength
Giriş

İşaretçiyi MQUZE. Güvenlik çıkışı çağrılırken, AgentBuffer içinde yer alan verilerin uzunluğu.

PMQVOID pAgentArabelleği
giriş/çıkış

Güvenlik Çıkışı çağrılırken, bu, ortak çıkıştan gönderilen herhangi bir iletiye işaret eder. MQCXP yapısında ExitResponse2 varsa, MQXR2_USE_AGENT_BUFFER işaret kümesi (varsayılan değer) varsa, bir Güvenlik Çıkışı 'nın gönderilmekte olan ileti verilerini göstermek için bu değiştirgeyi ayarlamaya gerek vardır.

PMQlong pExitBufferLength
giriş/çıkış

İşaretçiyi MQUZE. Bu parametre, bir Security Exit 'in ilk çağrısında 0 ile başlatılır ve döndürülen değer, güvenlik değiş tokası sırasında Güvenlik Çıkışı çağrıları arasında tutulur.

PMQPTR pExitBufferAddr
giriş/çıkış

Bu değiştirge, bir Security Exit 'in ilk çağrısında boş değerli bir işaretçi olarak başlatılır ve döndürülen değer, güvenlik değiş tokası sırasında Güvenlik Çıkışı çağrıları arasında tutulur. If the MQXR2_USE_EXIT_BUFFER flag is set in the ExitResponse2 in the MQCXP structure then a Security Exit needs to set this parameter to point to any message data being sent.

CLNTCONN/SVRCONN kanal çiftlerinde ve diğer kanal çiftlerinde tanımlanan güvenlik çıkışlarıyla ilgili davranış farklılıkları
Güvenlik çıkışları, tüm kanal tiplerinde tanımlanabilir. Ancak, CLNTCONN/SVRCONN kanal çiftlerinde tanımlanan güvenlik çıkışlarının işleyişi, diğer kanal çiftlerinde tanımlanan güvenlik çıkışlarından biraz farklıdır.

Bir CLNTCONN kanalındaki bir Güvenlik Çıkışı, iş ortağı SVRCONN çıkışa göre işlenmek üzere kanal tanımlamasındaki Uzak Kullanıcı Tanıtıcısını ya da SVRCONN Güvenlik Çıkışı tanımlanmadıysa ve SVRCONN ' un MCAUSER alanı ayarlanmadıysa OAM yetkilendirmesi olarak ayarlayabilir.

CLNTCONN Security Exit tanımlanmadıysa, kanal tanımlamasındaki Uzak Kullanıcı Tanıtıcısı istemci ortamından (boş olabilir) istemci ortamından bir kullanıcı kimliği olarak ayarlanır.

SSVRCONN Security Exit, MQXCC_OK ExitResponse değerini döndürdüğünde, CLNTCONN ve SVRCONN kanal çiftinde tanımlanan Güvenlik Exits ile SVRCONN kanal çiftinde tanımlanan bir güvenlik değiş tokası başarıyla tamamlanır. Diğer kanal çiftleri arasındaki bir güvenlik değiş tokası başarıyla tamamlanırsa, değiş tokuş işlemi başlatan Security Exit, MQXCC_OK için ExitResponse (ExitResponse) değerini döndürür.

Ancak, güvenlik değiş tokuşunu devam ettirmeye zorlamak için MQXCC_SEND_AND_REQUEST_SEC_MSG ExitResponse kodu kullanılabilir: MQXCC_SEND_AND_REQUEST_SEC_MSG çizelgesinin bir ExitResponse bir CLNTCONN ya da SVRCONN Security Exit tarafından döndürülürse, iş ortağı çıkışı bir güvenlik iletisi göndererek yanıt vermelidir (MQXCC_OK ya da boş bir yanıt değil) ya da kanal sonlandırılır. For Security Exits defined on other types of channel, an ExitResponse of MQXCC_OK returned in response to a MQXCC_SEND_AND_REQUEST_SEC_MSG from the partner Security Exit results in continuation of the security exchange as if a null response was returned and not in termination of the channel.

SSPI güvenlik çıkışı
WebSphere MQ for Windows , Security Services Programming Interface (SSPI) olanağını kullanarak WebSphere MQ kanalları için kimlik doğrulama sağlayan bir güvenlik çıkışı sağlar. The SSPI provides the integrated security facilities of Pencereler.

Bu güvenlik çıkışı, hem WebSphere MQ istemcisi hem de WebSphere MQ sunucusu için kullanılabilir.

Güvenlik paketleri security.dll ya da secur32.dlliçinden yüklenir. Bu DLL ' ler işletim sisteminiz ile birlikte sağlanır.

NTLM kimlik doğrulama hizmetleri kullanılarak, Windowsüzerinde tek yönlü kimlik doğrulaması sağlanır. Two-way authentication is provided on Pencereler 2000, using Kerberos authentication services.

Güvenlik çıkış programı kaynak ve nesne biçiminde sağlanır. Nesne kodunu olduğu gibi kullanabilir ya da kaynak kodu, kendi kullanıcı çıkış programlarınızı yaratmak için başlangıç noktası olarak kullanabilirsiniz. SSPI güvenlik çıkışımın nesne ya da kaynak kodunun kullanılmasına ilişkin ek bilgi için bkz. “ Windows sistemlerinde SSPI güvenlik çıkışısının kullanılması” sayfa 162

Kanal gönderme ve alma çıkış programları
Veri sıkıştırma ve açma gibi görevleri gerçekleştirmek için gönderme ve alma çıkışlarını kullanabilirsiniz. Gönderme ve alma çıkış programlarının art arda çalıştırılacağı bir listesini belirtebilirsiniz.

Kanal gönderme ve alma çıkış programları, MCA ' nın işlem çevriminde aşağıdaki yerlerde çağrılır:

- Gönderme ve alma çıkış programları, MCA başlatma sırasında kullanıma hazırlama ve MCA sonlandırmasında sona erdirme için çağrılır.
- Çıkış gönderme programı, bir ileti aktarımın gönderileceği uca bağlı olarak, kanalın bir ya da başka bir ucunda çağrılır ve bağlantı üzerinden bir iletim gönderilmeden hemen önce gönderilir. Not 4, ileti kanalları iletileri tek bir yöne gönderse de, çıkışlar neden her iki yönde de kullanılabilir olduğunu açıklar.
- Alma çıkış programı, bir ileti aktarımı için bir iletim alındığı sona erme durumuna bağlı olarak, kanalın bir ya da diğer ucunda çağrılır ve bağlantıdan hemen bir iletim alınır. Not 4, ileti kanalları iletileri tek bir yöne gönderse de, çıkışlar neden her iki yönde de kullanılabilir olduğunu açıklar.

Bir ileti aktarımı için birçok iletim olabilir ve bir ileti alan uçta ileti çıkışa ulaşmadan önce, gönderme ve alma çıkış programlarının birçok yinelemesi olabilir.

Kanal gönderme ve alma çıkış programları, iletişim bağlantısından gönderilen ya da alınan iletim verilerini içeren bir aracı arabelleğinden geçirilir. Çıkış programları göndermek için, arabelleğin ilk 8 baytı MCA tarafından kullanılmak üzere ayrılır ve değiştirilmemelidir. Program farklı bir arabellek döndürürse, bu ilk 8 byte yeni arabelleğde var olmalıdır. Çıkış programlarına sunulan verilerin biçimi tanımlı değil.

Çıkış programları gönderme ve alma yoluyla iyi bir yanıt kodu döndürülemelidir. Diğer herhangi bir yanıt MCA ' nın olağandışı bitmesine (olağandışı bitme) neden olur.

Not: Gönderme ya da alma çıkışından tutarlılık noktası içinde bir MQGET, MQPUT ya da MQPUT1 çağrısı yayınlanmayın.

Not:

1. Gönderme ve alma çıkışları genellikle çiftler halinde çalışılır. Örneğin, bir gönderme çıkışı, verileri sıkıştırabilir ve bir alma çıkışı sıkıştırabilir ya da bir gönderme çıkışı verileri şifreleyebilir ve bir alma çıkışı bunu çözebilir. Uygun kanalları tanımladığınızda, kanalın her iki ucu için uyumlu çıkış programlarının adlandırıldığından emin olun.
2. Kanal için sıkıştırma açıksa, çıkışlar sıkıştırılmış veriler iletilir.
3. Kanal gönderme ve alma çıkışları, uygulama verileri (örneğin, durum iletileri) dışındaki ileti kesimleri için çağrılabilir. Bunlar, başlatma iletişim kutusu sırasında çağrılmaz ya da güvenlik denetimi evresidir.
4. İleti kanalları iletileri tek bir yönde gönderse de, kalp atışları ve toplu iş işlemlerinin sonu gibi kanal denetimi verileri her iki yöne doğru akar ve bu çıkışlar her iki yönde de kullanılabilir. Ancak, ilk kanal başlatma veri akışlarından bazıları, çıkışlardan herhangi biri tarafından işlenmekten muaf tutulmakta.
5. Gönderme ve alma çıkışlarının sırayla çağrılabileceği durumlar vardır; örneğin, bir dizi çıkış programı çalıştırıyorsanız ya da güvenlik çıkışlarını çalıştırıyorsanız. Daha sonra, alma çıkışı veri işlemek için ilk çağrıldığında, ilgili gönderme çıkıştan geçmemiş veriler alabilir. Alma çıkışı işlemi az önce gerçekleştirdiyse, örneğin, açma işleminin gerekli olduğunu denetlemeden, açma işlemi beklenmeyen bir şekilde sonuçlanabilir.

   Gönderme ve alma çıkışlarınızı, alma çıkığı, aldığı verilerin ilgili gönderme çıkışıyla işlenip işlendiğini denetleyebilmesi için çıkış kodlarına gerek duyarsınız. Bunun için önerilen yol, çıkış programlarınızı kodlamak olmalıdır.

- Çıkış gönderme işlemi, işlemi gerçekleştirmeden önce, dokuzuncu bayt veri değerini 0 olarak ayarlar ve tüm verileri 1 bayt boyunca kaydırır. (İlk 8 byte, MCA tarafından kullanılmak üzere ayrılmıştır.)
- Alma çıkışı bayt 9 'da 0 olan verileri alırsa, veri gönderme çıkışından geldiğini bilir. 0 'ı kaldırır, tamamlayıcı işlemi gerçekleştirir ve sonuçta elde edilen verileri 1 bayt geriye kaydırır.
- Alma çıkışı, bayt 9 'da 0 dışında bir değer içeren verileri alırsa, gönderme çıkısının çalıştırılmadığını varsayar ve verileri arayanın geri göndermesine geri gönderir.

Güvenlik çıkışlarını kullanırken, kanal güvenlik çıkışı tarafından sona erdirilirse, ilgili alma çıkışı olmadan bir gönderme çıkışı çağrılabilecektir. Bu sorunun önüne geçmenin bir yolu da, güvenlik çıkışını MQCD.SecurityUserData ya da MQCD.SendUserDataiçinde bir işaret ayarlamak için kodlamak; örneğin, çıkış kanalı sona erdirmeye karar verince. Daha sonra, çıkış gönderme işleminin bu alanı denetlemesi ve yalnızca işaret ayarlanmadıysa verileri işlemesi gerekir. Bu denetim, verilerin gereksiz yere değiştirilmesine neden olmasını önler ve güvenlik çıkışı değiştirilmiş verileri aldıysa, oluşabilecek dönüştürme hatalarını önler.

Kanal gönderme çıkış programları-ayırma alanı
İletinin iletilmeden önce dönüştürülmesi için, gönderme ve alma çıkışlarını kullanabilirsiniz. Kanal gönderme çıkış programları, dönüştürme arabelleğindeki alanı ayırarak dönüştürmeyle ilgili kendi verilerini ekleyebilir.

Bu veriler, alma çıkış programı tarafından işlenir ve arabelleğinden kaldırılır. Örneğin, verileri şifrelemek ve şifre çözme için bir güvenlik anahtarı eklemek isteyebilirsiniz.

Alanı nasıl ayırdığınız ve nasıl kullanabildiğinizi

Başlatma işlemi için gönderme çıkış programı çağrıldığında, MQXCP ' nin *ExitSpace* alanını, ayrılacak bayt sayısına ayarlayın. Ayrıntılar için bkz. MQCXP . *ExitSpace* can be set only during initialization, that is when *ExitReason* has the value MQXR_INIT. Gönderme çıkışı iletilmeden hemen önce çağrıldığında, *ExitReason* MQXR_XMIT olarak ayarlanınca, *ExitSpace* byte iletim arabelleğindeki ayrıdır. *ExitSpace* , z/OSüzerinde desteklenmez.

Çıkış gönderme ihtiyacı, tüm ayrılmış alanı kullanmaz. *ExitSpace* byte 'tan az bir değer kullanabilir ya da iletim arabelleği dolu değilse, çıkış ayrılmış miktardan daha fazlasını kullanabilir. *ExitSpace*değerini ayarlarken, iletim arabelleğindeki ileti verileri için en az 1 KB ' lik bir değer bırakmanız gerekir. Ayrılmış alan büyük miktarda veri için kullanılıyorsa, kanal başarımı etkilenebilir.

kanal d ' nin giriş sonunda ne olur?

Kanal alma çıkış programlarının, ilgili gönderme çıkışlarıyla uyumlu olması için ayarlanması gerekir. Alma çıkışları, ayrılmış alanda bayt sayısını bilmeli ve bu alandaki verileri kaldırmalıdır.

Birden çok gönderme çıkışı

Gönderme ve alma çıkış programlarının art arda çalıştırılacağı bir listesini belirtebilirsiniz. WebSphere MQ , tüm gönderme çıkışları tarafından ayrılmış alan için toplam tutar sağlar. İletim arabelleğindeki ileti verileri için bu toplam alan en az 1 KB ' lik bir değer bırakmalıdır.

Aşağıdaki örnekte, üç gönderme çıkışı için alanın ayrıldığı sırayla nasıl ayrıldığı gösterilmektedir:

1. Başlatma için çağrıldığında:
   - Çıkış yollarına 1 KB ' lik rezerv gönderin.
   - Çıkış B 'ye 2 KB' yi gönderin.
   - C yedek çıkış 3 KB ' yi gönderin.
2. İletim büyüklüğü üst sınırı 32 KB 'dir ve kullanıcı verileri 5 KB uzunluğunda olur.
3. A çıkışı 5 KB veri ile çağrılır; 27 KB ' ye kadar kullanılabilir, çünkü 5 KB, B ve C çıkışlarına ayrılmıştır. Çıkış A ' dan çıkış 1 KB, ayırmış olduğu miktar.

4. B çıkışı 6 KB veri ile çağrılır; 29 KB ' ye kadar kullanılabilir, çünkü C çıkışı için 3 KB ayrılmıştır. Çıkış B, ayrılmış 2 KB 'den az olan 1 KB' yi ekler.
5. Exit C, 7 KB veri ile çağrılır; 32 KB ' ye kadar kullanılabilir. C çıkışı, ayrılmış 3 KB 'den fazla 10K' yı ekler. Bu tutar, toplam veri miktarı 17 KB 'nin 32 KB' den az olduğu için geçerlidir.

Kanal ileti çıkış programları
Kanal ileti çıkışını kullanarak, şifreleme, gelen kullanıcı kimliklerinin doğrulanması ya da ikamesi, ileti verileri dönüştürme, günlük kaydı ve başvuru iletisi işleme gibi görevleri gerçekleştirmek için kullanabilirsiniz. Art arda çalıştırılacak ileti çıkış programlarının bir listesini belirtebilirsiniz.

Kanal ileti çıkış programları MCA ' nın işlem çevriminde aşağıdaki yerlerde çağrılır:

- MCA başlatma ve sonlandırma sırasında
- MCA gönderdikten hemen sonra bir MQGET çağrısı yayınlandıktan sonra
- Alma MCA ' nın bir MQPUT çağrısı yayınlamadan önce

İleti çıkışı, kuyruktan alınan iletim kuyruğu üstbilgisini, MQXQH ' yi ve uygulama ileti metnini içeren bir aracı arabelleğinden geçirilir. (MQXQH biçimi MQXQH' de verilmiştir.) Başvuru iletileri kullanıyorsanız; yalnızca, gönderilecek başka bir nesneyi işaret eden bir üstbilgi içeren iletiler, ileti çıkışı üstbilgiyi tanır, MQRMH ' dir. Nesneyi tanımlar, uygun şekilde alır, uygun şekilde üstbilgiye ekler ve alma MCA 'ya iletilmesi için MCA' ya iletir. Alıcı MCA ' da, başka bir ileti çıkışı bu iletinin bir başvuru iletisi olduğunu algılar, nesneyi çeker ve üstbilgiyi hedef kuyruğa aktarır. Başvuru iletilerine ve bunları işleten bazı örnek ileti çıkışlarına ilişkin ek bilgi için “Başvuru iletileri” sayfa 254 ve “Başvuru İletisi örneklerinin çalıştırılması” sayfa 135 başlıklı konuya bakın.

İleti çıkışları aşağıdaki yanıtları döndürebilir:

- İletiyi gönderin (GET çıkışı). Çıkış, çıkış tarafından değiştirilmiş olabilir. (Bu, MQXCC_OK değerini döndürür.)
- İletiyi kuyruğa koyun (PUT çıkışı). Çıkış, çıkış tarafından değiştirilmiş olabilir. (Bu, MQXCC_OK değerini döndürür.)
- İletiyi işlemeyin. İleti, MCA tarafından gönderilen ileti kuyruğunda (teslim edilmemiş ileti kuyruğu) yerleştirilir.
- Kanalı kapatın.
- MCA ' nın olağandışı sona ermesine neden olan dönüş kodu hatalı.

Not:

1. İleti, aktarılan her tam ileti için bir kez çağrılır ve ileti parçalara bölünse bile.
2. UNIX sistemlerinde, herhangi bir nedenle kullanıcı kimliklerinin küçük harfe otomatik olarak dönüştürülmesinin çalışmaması için bir ileti çıkışı sağlanmaz. Bkz. UNIX and Linux sistemlerindeki nesnelerin güvenliği.
3. Bir çıkış MCA ' nın kendisi ile aynı iş parçacığıda çalışır. Aynı çalışma birimi (UOW) içinde aynı bağlantı tanıtıcısını kullandığı için aynı çalışma birimi (UOW) içinde de çalışır. Bu nedenle, eşitleme noktası altında yapılan tüm çağrılar, toplu işin sonundaki kanaldan kesinleştirilir ya da yedeklenir. Örneğin, bir kanal ileti çıkışı programı, başka bir kanala bildirim iletileri gönderebilir ve bu iletiler, yalnızca özgün iletiyi içeren toplu iş kesinleştirildiğinde kuyruğa kesinleşmektedir.

   Bu nedenle, bir kanal iletisi çıkış programından Sync Point MQI çağrılması mümkün olabilir.

İleti çıkışı dışında ileti dönüştürme
İleti çıkışını çağırmadan önce, alıcı MCA ileti üzerinde bazı dönüştürmeler gerçekleştirir. Bu konuda, dönüştürmeleri gerçekleştirmek için kullanılan algoritmalar açıklanmaktadır.

Hangi üstbilgilerin işlendiği

İleti çıkışı çağrılmadan önce, alıcının MCA ' sında bir dönüştürme yordamı çalıştırılır. Dönüştürme yordamı, iletinin başlangıcındaki MQXQH üstbilgisiyle başlar. Dönüştürme yordamı, MQXQH ' yi izleyen zincirleme üstbilgiler yoluyla, gerektiğinde dönüştürme işlemini gerçekleştirmektedir. Zincirleme üstbilgiler, alıcının ileti çıkışa geçirilen MQCXP verilerinin HeaderLength değiştirgesinde yer alan göreli konumun ötesini genişletebilirler. Aşağıdaki üstbilgiler yerinde dönüştürüldü:

- MQXQH (biçim adı "MQXMIT ")
- MQMD (bu üstbilgi MQXQH ' nin bir parçası ve biçim adı yok)
- MQMDE (biçim adı "MQHMDE ")
- MQDH (biçim adı "MQHDIST ")
- MQWIH (biçim adı "MQHWIH ")

Aşağıdaki üstbilgiler dönüştürülmez, ancak MCA zincirleme üstbilgileri işlemeye devam ettikçe devreye girilir:

- MQDLH (biçim adı "MQDEAD ")
- 'MQH' karakteriyle başlayan, biçim adları olan üstbilgiler (örneğin "MQHRF ") bu, başka bir şekilde belirtilmeyen

Üstbilgilerin nasıl işlendiği

Her WebSphere MQ üstbilgisinin Format parametresi MCA tarafından okunur. Biçim parametresi, üstbilgi içindeki 8 byte 'tır (bir ad içeren 8 tek baytlık karakter).

Daha sonra, MCA, her bir üstbilginin adını belirtilen tipte olarak yorumlayarak verileri yorumlar. Biçim, WebSphere MQ veri dönüştürmesi için uygun bir üstbilgi tipinin adıdır, dönüştürülemektedir. MQ dışı verileri gösteren başka bir ad ise (örneğin, MQFMT_NONE ya da MQFMT_STRING gibi), MCA üstbilgileri işlemeyi durdurur.

MQCXP HeaderLengthnedir?

İleti çıkışa sağlanan MQCXP verilerindeki HeaderLength değiştirgesi, iletinin başlangıcındaki MQXQH (MQMD ' yi içerir), MQMDE ve MQDH üstbilgilerini içeren toplam uzunluktır. Bu üstbilgiler 'Biçim' adları ve uzunlukları kullanılarak zincirlenir.

MQWIH

Zincirleme üstbilgiler, kullanıcı verileri alanına HeaderLength alanının ötesine kadar uzayabilir. MQWIH üstbilgisi (varsa), HeaderLength(HeaderLength) altında görüntülenen üstbilgilerden biridir.

Zincirli üstbilgilerde bir MQWIH üstbilgisi varsa, alıcının ileti çıkışı çağrılmadan önce bu üstbilgi yerine dönüştürülür.

Kanal iletisi yeniden deneme çıkış programı
Kanal iletisi-yeniden deneme çıkışı, hedef kuyruğu açma girişimi başarısız olduğunda çağrılır. Hangi koşullar altında yeniden deneneceğini, kaç kez yeniden deneneceğini ve ne sıklıkta yeniden deneneceğini belirlemek için çıkışı kullanabilirsiniz.

Bu çıkış, MCA başlatma ve sonlandırma sırasında kanalın giriş bitişindeki çağrıdır.

Kanal iletisine yeniden deneme çıkışı, iletim kuyruğu üstbilgisini içeren bir aracı arabelleğinden, MQXQH ve kuyruktan alınan uygulama iletisi metninden geçirilir. MQXQH biçimi Overview for MQXQH(MQXQH için genel bakış) biçiminde verilir.

Çıkış, tüm neden kodları için çağrılır; çıkış, MCA ' nın kaç kez yeniden denemesini, kaç kez ve hangi aralıklarla yeniden denemesini istediğini belirler. (Kanal tanımlandığında, iletinin yeniden deneme sayısı, MQCD ' deki çıkışa geçirilir, ancak çıkış bu değeri yoksayabilir.)

MQCXP ' deki MsgRetrySayı alanı, çıkış çağrıldığında MCA tarafından artırılır ve çıkış, MQCXP ya da MQXCC_SUPPRESS_FUNCTION MsgRetryAralığı alanında bekleme süresiyle MQXCC_OK değerini döndürür. Çıkış, MQCXP ' nin ExitResponse alanında MQXCC_SUPPRESS_FUNCTION döndürünceye kadar süresiz olarak devam eder. Bu tamamlanma kodlarına ilişkin MCA tarafından alınan işlemle ilgili bilgi için MQCXP başlıklı konuya bakın.

Yeniden denemelerin tümü başarısız olursa, ileti, ölü-mektup kuyruğuna yazılır. Kullanılabilir bir ölü harf kuyruğu yoksa, kanal durur.

Bir kanal için ileti yeniden deneme çıkışı tanımlamadıysanız ve geçici olarak büyük olasılıkla geçici (örneğin MQRC_Q_FULL) gerçekleştirilmezse, MCA iletiyi kullanır-yeniden deneme sayısı ve ileti-yeniden deneme aralıkları kanal tanımlandığında ayarlanır. Hata daha kalıcı bir nitelimeyse ve bunu işlemek için bir çıkış programı tanımlamadıysanız, ileti ölü-mektup kuyruğuna yazılır.

Kanal otomatik tanımlama çıkış programı
The channel auto-definition exit can be used when a request is received to start a receiver or server-connection channel but no definition for that channel exists (not for WebSphere MQ for z/OS). Ayrıca, tüm altyapılarda, bir kanal yönetim ortamı için tanım değişikliğine izin vermek üzere, küme gönderici ve küme alıcı kanallarına ilişkin tüm altyapılarda da çağrılabilir.

Kanal otomatik tanımlama çıkışı, bir alıcı ya da sunucu bağlantısı kanalını başlatmak için bir istek alındığında z/OS dışında tüm platformlarda çağrılabilir, ancak kanal tanımlaması yok. Bunu, otomatik olarak tanımlanmış bir alıcı ya da sunucu bağlantısı kanalı olan SYSTEM.AUTO.RECEIVERya da SYSTEM.AUTO.SVRCON. Kanal tanımlamalarının otomatik olarak nasıl yaratılabileceğiyle ilgili açıklamalar için Kanalların hazırlanması başlıklı konuya bakın.

Kanal otomatik tanımlama çıkışı, bir küme gönderici kanalı başlatmak için bir istek alındığında da çağrılabilir. Bu kanal, bu kanal yönetim ortamı için tanımlama değişikliğine izin vermek üzere, küme gönderici ve küme alıcı kanalları için çağrılabilir. Bu durumda, çıkış, z/OSiçin WebSphere MQ için de geçerlidir. Çıkış adlarının farklı platformlarda farklı biçimlerde olması nedeniyle, kanal otomatik tanımlama çıkışımın ortak bir kullanımı, ileti çıkışlarının (MSGEXIT, RCPEXIT, SCYEXIT ve SENDEXIT) adlarını değiştirmemektedir. If no channel auto-definition exit is specified, the default behavior on z/OS is to examine a distributed exit name of the form *[path]/libraryname(function)* and take up to eight chars of function, if present, or libraryname. On z/OS, a channel auto-definition exit program must alter the fields addressed by MsgExitPtr, MsgUserDataPtr, SendExitPtr, SendUserDataPtr, ReceiveExitPtr, and ReceiveUserDataPtr, rather than the MsgExit, MsgUserData, SendExit, SendUserData, ReceiveExit and ReceiveUserData fields themselves.

Ek bilgi için Kanalların otomatik olarak tanımlanmasıbaşlıklı konuya bakın.

Diğer kanal çıkışlarında olduğu gibi, parametre listesi şöyle olur:

```
MQ_CHANNEL_AUTO_DEF_EXIT (ChannelExitParms, ChannelDefinition)
```

`ChannelExitParms` , MQCXPiçinde açıklanır. `ChannelDefinition` , MQCD' de açıklanmaktadır.

MQCD, çıkış tarafından değiştirilmezse, varsayılan kanal tanımında kullanılan değerleri içerir. Çıkış, alanların yalnızca bir alt kümesini değiştirebilir; bkz. MQ_CHANNEL_AUTO_DEF_EXIT. Ancak, diğer alanları değiştirme girişimi bir hataya neden olmaz.

Kanal otomatik tanımlama çıkışı, MQXCC_OK ya da MQXCC_SUPPRESS_FUNCTION yanıt verir. Bu yanıtlardan hiçbiri döndürülmezse, MCA işlemi, MQXcc_suppress_function döndürüldüğü halde işlenmeye devam eder. Yani, otomatik tanımlama iptal edilir, yeni kanal tanımlaması yaratılamaz ve kanal başlayamaz.

Windows, UNIX and Linux sistemlerinde kanal çıkış programlarının derlenmesi

Windows, UNIX and Linux sistemleri için kanal çıkış programlarını derlemenize yardımcı olması için aşağıdaki örnekleri kullanın.

Pencereler

Windows

Windows üzerinde kanal çıkış programları için derleyici ve linker komutu:

```
cl.exe /Ic:\mqm\tools\c\include /nologo /c myexit.c
link.exe /nologo /dll myexit.obj /def:myexit.def /out:myexit.dll
```

UNIX ve Linux sistemleri

Linux UNIX

Bu örneklerde `exit` , kitaplık adı ve `ChannelExit` , işlev adıdır. On AIX the export file is called `exit.exp`. These names are used by the channel definition to reference the exit program using the format described in MQCD-kanal tanımlaması. Ayrıca, DEFINE CHANNEL komutunun MSGEXIT parametresine de bakın.

Sample compiler and linker commands for channel exits on AIX:

```
$ xlc_r -q64 -e MQStart -bE:exit.exp -bM:SRE -o /var/mqm/exits64/exit
exit.c -I/usr/mqm/inc
```

HP-UXüzerindeki kanal çıkışları için örnek derleyici ve linker komutları

```
$ c89 +DD64 +z -c -D_HPUX_SOURCE -o exit.o exit.c -I/opt/mqm/inc
$ ld -b exit.o +ee MQStart +ee ChannelExit -o
/var/mqm/exits64/exit -L/usr/lib/pa20_64 -lpthread
$ rm exit.o
```

Kuyruk yöneticisinin 32 bit olduğu Linux platformlarında kanal çıkışlarına ilişkin örnek derleyici ve bağlantı komutları.

```
$ gcc -shared -fPIC -o /var/mqm/exits/exit exit.c -I/opt/mqm/inc
```

Kuyruk yöneticisinin 64 bit olduğu Linux platformlarında kanal çıkışlarına ilişkin örnek derleyici ve bağlantı komutları.

```
$ gcc -m64 -shared -fPIC -o /var/mqm/exits64/exit exit.c -I/opt/mqm/inc
```

Solaris üzerindeki kanal çıkışlarına ilişkin örnek derleyici ve bağlantı komutları:

```
$ cc -xarch=v9 -mt -G -o /var/mqm/exits64/exit exit.c -I/opt/mqm/inc
-R/usr/lib/64 -lsocket -lnsl -ldl
```

İstemcide, 32 bit ya da 64 bit çıkışı kullanılabilir. Bu çıkış mqic_r ile bağlantılandırılmalıdır.

AIXüzerinde, IBM WebSphere MQ tarafından çağrılan tüm işlevler dışa aktarılmalıdır. Bu make dosyası için örnek bir dışa aktarma dosyası:

```
#!
channelExit
MQStart
```

Kanal çıkışlarının yapılandırılması

Kanal çıkışını aramak için, kanal tanımını kanal tanımında adlanmanız gerekir.

Kanal çıkışlarında kanal çıkışlarının adlandırılması gerekir. Kanalları ilk kez tanımladığınızda bu adlandırma işlemi yapabilir ya da daha sonra, MQSC komutu ALTER CHANNEL 'ı kullanarak bilgileri de ekleyebilirsiniz. Ayrıca, ilgili MQCD kanalı veri yapısında kanal çıkış adlarını da verebilirsiniz. Çıkış adının biçimi, IBM WebSphere MQ altyapınıza bağlıdır; bilgi için MQCD ya da Script (MQSC) Commands belgesine bakın.

Kanal tanımlaması kullanıcı çıkışı programı adı içermiyorsa, kullanıcı çıkışı çağrılmaz.

Kanal otomatik tanımlama çıkışı, tek kanalda değil, kuyruk yöneticisinin özelidir. Bu çıkışa çağrılması için, kuyruk yöneticisi tanımlamasında adlandırılması gerekir. Bir kuyruk yöneticisi tanımlamasını değiştirmek için, MQSC komutu ALTER QMGR komutunu kullanın.

Veri dönüştürme çıkışları yazılıyor

Bu konu derlemi, veri dönüştürme çıkışlarının nasıl yazılacağı ile ilgili bilgileri içerir.

Not: VSE/ESAiçin MQSeries içinde desteklenmez.

Bir MQPUT uyguladığınızda, uygulamanız iletinin ileti tanımlayıcısını (MQMD) yaratır. WebSphere MQ 'un yarattığı altyapıdan bağımsız olarak, MQMD' nin içeriğini anlayabilmesi için, sistem tarafından otomatik olarak dönüştürülür.

Ancak, uygulama verileri otomatik olarak dönüştürülmez. Karakter verileri, *CodedCharSetId* ve *Encoding* alanlarının farklı olduğu platformlar arasında değiş tokuş ediliyorsa, örneğin, ASCII-EBCDIC arasında, uygulamanın iletiyi dönüştürmesi için bir düzenleme işlemi ayarlamalıdır. Uygulama verileri dönüştürme, kuyruk yöneticisinin kendisi ya da *veri-dönüştürme çıkışı*olarak adlandırılan bir kullanıcı çıkış programı tarafından gerçekleştirilebilir. Uygulama verileri yerleşik biçimlerden biriyse (MQFMT_STRING gibi), kuyruk yöneticisi yerleşik dönüştürme yordamlarından birini kullanarak veri dönüştürmeyi kendisi gerçekleştirebilir. Bu konu, uygulama verilerinin yerleşik bir biçimde olmadığı durumlarda WebSphere MQ ' un sağladığı veri dönüştürme çıkış tesisine ilişkin bilgileri içerir.

Denetim, bir MQGET çağrısı sırasında veri dönüştürme çıkışa geçirilebilir. Bu, son hedefe ulaşmadan önce farklı platformlara dönüştürülmeyi önler. Ancak, son hedef, MQGET üzerinde veri dönüştürmeyi desteklemeyen bir altyapıysa, verileri son hedefine gönderen gönderici kanalıyla CONVERT (YES) belirtmelisiniz. This ensures that WebSphere MQ converts the data during transmission. Bu durumda, veri dönüştürme çıkışınızın, gönderen kanalının tanımlandığı sistemde bulunması gerekir.

MQGET çağrısı doğrudan uygulama tarafından verilir. MQMD ' deki *CodedCharSetId* ve *Encoding* alanlarını, gerekli karakter takımı ve kodlamaya ayarlayın. Uygulamanız kuyruk yöneticisi olarak aynı karakter kümesini ve kodlamayı kullanıyorsa, *CodedCharSetId* seçeneğini, MQCCSI_Q_MGR ve *Encoding* olarak MQENC_NATIVE olarak ayarlayın. MQGET çağrısının tamamlanmasından sonra, bu alanlar döndürülen ileti verilerine uygun değerlere sahiptir. Bu değerler, dönüştürme başarılı olmadıysa, gereken değerlerden farklı olabilir. Uygulamanızın bu alanları, her MQGET çağrısından önce gerekli olan değerlere sıfırlaması gerekir.

Çağrılacak veri dönüştürme çıkışı için gerekli olan koşullar, MQGET' de MQGET çağrısı için tanımlandır.

Veri dönüştürme çıkışa ve ayrıntılı kullanım notlarına geçirilen parametrelere ilişkin açıklamalar için, MQ_DATA_CONV_EXIT çağrısına ve MQDXP yapısına ilişkin Veri dönüştürmesi konusuna bakın.

Farklı makine kodlamaları ve CCSID ' ler arasında uygulama verilerini dönüştüren programlar, WebSphere MQ veri dönüştürme arabirimine (DCI) uygun olmalıdır.

Bazı iletiler kuyruk yöneticisinden geçemeyebileceğinden, çok hedefli istemciler, API çıkışlar ve veri dönüştürme çıkışlarının istemci tarafında çalışabilmesi için bu çıkışlar istemcide çalışır. Aşağıdaki kitaplıklar, sunucu paketlerinin yanı sıra, istemci paketlerinin bir parçası olarak da yer alıyor:

Çizelge 56. İstemci ve sunucu paketlerinde bulunan kitaplıklar

İşletim sistemi	**Kitaplıklar**
Pencereler	32 bit & 64 bit: mqm.dll & mqm.pdb
Linux & HP-UX	32 bit ve 64 bit: libmqm.so & libmqm_r.so
AIX	32 bit & 64 bit: libmqm.a & libmqm_r.a
Solaris	32 bit ve 64 bit: libmqm.so

Veri dönüştürme çıkışı çağrılıyor

Veri dönüştürme çıkışı, bir MQGET çağrısının işlenmesi sırasında denetimi alan, kullanıcı tarafından yazılmış bir çıkışıdır.

Aşağıdaki öğe doğruysa, çıkış çağrılır:

- MQGET çağrısında MQGMO_CONVERT seçeneği belirtildi.
- İleti verilerinin bazıları ya da tümü istenen karakter kümesinde ya da kodlamada değil.
- İletiyle ilişkilendirilen MQMD yapısındaki *Format* alanı MQFMT_NONE değil.
- MQGET çağrısında belirtilen *BufferLength* sıfır değil.
- İleti verisi uzunluğu sıfır değil.
- İleti, kullanıcı tanımlı bir biçime sahip verileri içerir. Kullanıcı tanımlı biçim, iletinin tamamını kaplayabilir ya da öncesinde bir ya da daha çok yerleşik biçim olabilir. Örneğin, kullanıcı tanımlı biçimden önce bir MQFMT_DEAD_LETTER_HEADER biçiminden önce olabilir. Çıkış, yalnızca kullanıcı tanımlı biçimi dönüştürmek için çağrılır; kuyruk yöneticisi, kullanıcı tanımlı biçimden önce gelen tüm yerleşik biçimleri dönüştürür.

  Yerleşik bir biçimi dönüştürmek için kullanıcı tarafından yazılan bir çıkış da çağrılabilir, ancak bu yalnızca yerleşik dönüştürme yordamları yerleşik biçimi başarılı bir şekilde dönüştüremiyorsa gerçekleşir.

MQ_DATA_CONV_EXIT' ta MQ_DATA_CONV_EXIT çağrısının kullanım notlarında tam olarak açıklanan başka koşullar da vardır.

MQGET çağrısına ilişkin ayrıntılar için MQGET konusuna bakın. Veri dönüştürme çıkışları, MQXCNVC dışındaki MQI çağrılarını kullanamaz.

Bir uygulama kuyruk yöneticisine bağlı uygulamadan bu yana *Format* ' u kullanan ilk iletiyi almayı denediğinde çıkışa ilişkin yeni bir kopya yüklenir. Kuyruk yöneticisi önceden yüklenmiş bir kopyayı atmışsa, başka zamanlarda yeni bir kopya da yüklenebilir.

Veri dönüştürme çıkışı, MQGET çağrısını yayınlayan programdan benzer bir ortamda çalışır. Kullanıcı uygulamalarının yanı sıra, program, ileti dönüştürmeyi desteklemeyen bir hedef kuyruk yöneticisine ileti gönderirken bir MCA (ileti kanalı aracısı) olabilir. Ortam, geçerli olduğu yerlerde, adres alanını ve kullanıcı profilini içerir. Çıkış, kuyruk yöneticisinin ortamında çalışmadığından, kuyruk yöneticisinin bütünlüğünü tehlikeye atmaz.

UNIX and Linux sistemlerinde WebSphere MQ için veri dönüştürme çıkışı yazılıyor

UNIX and Linux sistemlerinde WebSphere MQ için veri dönüştürme çıkış programları yazılırken dikkate alınacak adımlar hakkında bilgi.

Aşağıdaki adımları izleyin:

1. İleti biçiminizi adlayın. Ad, MQMD ' nin *Format* alanına sığmalı ve büyük harfli olmalıdır; örneğin, MYFORMAT. *Format* adının başında boşluk bulunmamalıdır. Sondaki boşluklar yok sayılır. *Format* yalnızca sekiz karakter uzunluğunda olduğu için, nesnenin adı en çok sekiz boşluk karakteri olmayan karakterlere sahip olmalıdır. Bir iletiyi her gönderdiğinizde bu adı kullanmayı unutmayın.

   Veri dönüştürme çıkışı bir iş parçacıklı bir ortamda kullanılırsa, yüklenebilir nesnenin yivli bir sürüm olduğunu belirtmek için _r tarafından izlenmelidir.
2. İletinizi göstermek için bir yapı oluşturun. Bir örnek için Geçerli sözdizimi konusuna bakın.
3. Veri dönüştürme çıkışınız için bir kod parçası oluşturmak üzere bu yapıyı `crtmqcvx` komutu aracılığıyla çalıştırın.

   `crtmqcvx` komutu tarafından oluşturulan işlevler, tüm yapıların paketlenmiş olduğunu varsayan makroları kullanır; bu durumda, bu makroların hata durumunda olması gerekir.
4. Belirtilen çatı kaynağı dosyasını kopyalayın, bu dosyayı “1” sayfa 398adımında ayarladığınız ileti biçiminizin adıyla yeniden adlandırın. İskelet kaynak dosyası ve kopyası salt okunurdur.

   The skeleton source file is called amqsvfc0.c.

5. On WebSphere MQ for AIX, a skeleton export file called amqsvfc.exp is also supplied. Bu dosyayı kopyalayın ve MYFORMAT.EXPolarak yeniden adlandırın.
6. İskelet, *MQ_INSTALLATION_PATH*/inc dizininde örnek bir üstbilgi dosyası (amqsvmha.h) içerir; burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder. Bu dosyayı almak için, içerme yolunuzun bu dizini gösterdiğinden emin olun.

   amqsvmha.h dosyası, `crtmqcvx` komutu tarafından oluşturulan kod tarafından kullanılan makroları içerir. Dönüştürülecek yapı karakter verisi içeriyorsa, bu makrolar MQXCNVC ' yi çağırır.
7. Kaynak dosyada aşağıdaki açıklama kutularını bulun ve anlatıldığı gibi kodu ekleyin:
   a. Kaynak dosyanın sonuna doğru, bir yorum kutusu şununla başlar:

   ```
   /* Insert the functions produced by the data-conversion exit */
   ```

   Burada, “3” sayfa 398adımında oluşturulan kod parçasını ekleyin.
   b. Kaynak dosyanın ortasındaki bir yorum kutusu şununla başlar:

   ```
   /* Insert calls to the code fragments to convert the format's */
   ```

   Bunu, `Converttag`STRUCTişlevine ilişkin açıklama satırı yapılan bir çağrıyı izlemektedir.

   İşleve ilişkin adı, “7.a” sayfa 399adımında eklediğiniz işlevin adına çevirin. İşlevi etkinleştirmek için açıklama karakterlerini kaldırın. Birden çok işlev varsa, bunların her biri için çağrı yaratın.
   c. Kaynak dosyanın başlangıcındaki bir açıklama kutusu şununla başlar:

   ```
   /* Insert the function prototypes for the functions produced by */
   ```

   Burada, yukarıdaki “3” sayfa 398 adımında eklenen işlevlere ilişkin işlev prototip deyimlerini ekleyin.
8. Giriş noktası olarak MQStart 'ı kullanarak, çıkışınızı paylaşılan kitaplık olarak derleyin. Bunu yapmak için bkz. “UNIX and Linux sistemlerindeki veri dönüştürme çıkışlarının derlenmesi” sayfa 399.
9. Çıkışın çıkış dizinine yerlesin. Varsayılan çıkış dizini, 32 bit sistemler için `/var/mqm/exits` ve 64 bit sistemler için `/var/mqm/exits64`' dir. Bu dizinleri qm.ini ya da mqclient.ini dosyasında değiştirebilirsiniz. Bu yol, her bir kuyruk yöneticisi için ayarlanabilir ve çıkış yalnızca o yol ya da yollardaki aralara ararlıdır.

Not:

1. `crtmqcvx` paketlenmiş yapıları kullanıyorsa, tüm WebSphere MQ uygulamaları bu şekilde derlenmelidir.
2. Veri dönüştürme çıkış programları yeniden girişli olmalıdır.
3. MQXCNVC, bir veri dönüştürme işleminden çıkarılabilen *yalnızca* MQI çağrısıdır.

UNIX and Linux sistemlerindeki veri dönüştürme çıkışlarının derlenmesi
UNIX and Linux sistemlerinde veri dönüştürme çıkışısının nasıl derleneceği örnekler.

Tüm altyapılarda, modüle giriş noktası MQStart 'tır.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

AIX

Aşağıdaki komutlardan birini vererek çıkış kaynak kodunu derleyin:

32 bit uygulamalar

İş parçacıklı olmayan

```
cc -e MQStart -bE:MYFORMAT.exp -bM:SRE -o /var/mqm/exits/MYFORMAT \
    MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
xlc_r -e MQStart -bE:MYFORMAT.exp -bM:SRE -o /var/mqm/exits/MYFORMAT_r \
    MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

64 bit uygulamalar

İş parçacıklı olmayan

```
cc -q64 -e MQStart -bE:MYFORMAT.exp -bM:SRE -o /var/mqm/exits64/MYFORMAT \
    MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
xlc_r -q64 -e MQStart -bE:MYFORMAT.exp -bM:SRE -o /var/mqm/exits64/MYFORMAT_r \
    MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

HP-UX Itanium platformu

Aşağıdaki komut kümelerinden birini yayınlayarak çıkış kaynak kodunu derleyin ve bağlayın:

32 bit uygulamalar

İş parçacıklı olmayan

Çıkış kaynak kodunu derleyin:

```
c89 +e +z -c -D_HPUX_SOURCE -o MYFORMAT.o MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

Çıkış nesnesini bağla:

```
ld +b: -b MYFORMAT.o +ee MQStart -o  \
   /var/mqm/exits/MYFORMAT -L/usr/lib/hpux32
rm MYFORMAT.o
```

İş parçacıklı

Çıkış kaynak kodunu derleyin:

```
c89 +e +z -c -D_HPUX_SOURCE -o MYFORMAT.o MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

Çıkış nesnesini bağla:

```
ld +b: -b MYFORMAT.o +ee MQStart -o  \
   /var/mqm/exits/MYFORMAT_r -L/usr/lib/hpux32 \
   -lpthread
rm MYFORMAT.o
```

64 bit uygulamalar

İş parçacıklı olmayan

Çıkış kaynak kodunu derleyin:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o MYFORMAT.o MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

Çıkış nesnesini bağla:

```
ld -b MYFORMAT.o +ee MQStart \
    -o  /var/mqm/exits64/MYFORMAT \
    -L/usr/lib/hpux64
rm MYFORMAT.o
```

İş parçacıklı

Çıkış kaynak kodunu derleyin:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o MYFORMAT.o MYFORMAT.c -IMQ_INSTALLATION_PATH/inc
```

Çıkış nesnesini bağla:

```
ld -b MYFORMAT.o +ee MQStart  \
    -o  /var/mqm/exits64/MYFORMAT_r  \
    -L/usr/lib/hpux64 -lpthread
rm MYFORMAT.o
```

Linux

Aşağıdaki komutlardan birini vererek çıkış kaynak kodunu derleyin:

31 bit uygulamalar

İş parçacıklı olmayan

```
 gcc -m31 -shared -fPIC -o /var/mqm/exits/MYFORMAT MYFORMAT.c \
      -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
gcc -m31 -shared -fPIC -o /var/mqm/exits/MYFORMAT_r MYFORMAT.c
     -IMQ_INSTALLATION_PATH/inc
```

32 bit uygulamalar

İş parçacıklı olmayan

```
 gcc -m32 -shared -fPIC -o /var/mqm/exits/MYFORMAT MYFORMAT.c
      -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/MYFORMAT_r MYFORMAT.c
     -IMQ_INSTALLATION_PATH/inc
```

64 bit uygulamalar

İş parçacıklı olmayan

```
 gcc -m64 -shared -fPIC -o /var/mqm/exits64/MYFORMAT MYFORMAT.c
      -IMQ_INSTALLATION_PATH/inc
```

İş parçacıklı

```
 gcc -m64 -shared -fPIC -o /var/mqm/exits64/MYFORMAT_r MYFORMAT.c
      -IMQ_INSTALLATION_PATH/inc
```

Solaris

Aşağıdaki komutlardan birini vererek çıkış kaynak kodunu derleyin:

32 bit uygulamalar

SPARC altyapısı

```
cc -xarch=v8plus -KPIC -mt -G -o /var/mqm/exits/MYFORMAT \
   MYFORMAT.c -IMQ_INSTALLATION_PATH/inc -R/usr/lib/32 -lsocket -lnsl -ldl
```

x86-64 platformu

```
cc -xarch=386 -KPIC -mt -G -o /var/mqm/exits/MYFORMAT \
   MYFORMAT.c -IMQ_INSTALLATION_PATH/inc -R/usr/lib/32 -lsocket -lnsl -ldl
```

64 bit uygulamalar

SPARC altyapısı

```
cc -xarch=v9 -KPIC -mt -G -o /var/mqm/exits64/MYFORMAT \
   MYFORMAT.c -IMQ_INSTALLATION_PATH/inc -R/usr/lib/64 -lsocket -lnsl -ldl
```

x86-64 platformu

```
cc -xarch=amd64 -KPIC -mt -G -o /var/mqm/exits64/MYFORMAT \
   MYFORMAT.c -IMQ_INSTALLATION_PATH/inc -R/usr/lib/64  -lsocket -lnsl -ldl
```

WebSphere MQ for Windowsiçin veri dönüştürme çıkışı yazılıyor

WebSphere MQ for Windowsiçin veri dönüştürme çıkış programları yazılırken dikkate alınacak adımlar hakkında bilgi.

Aşağıdaki adımları izleyin:

1. İleti biçiminizi adlayın. Adın, MQMD ' nin *Format* alanına sığması gerekir. *Format* adının başında boşluk bulunmamalıdır. Sondaki boşluklar yok sayılır. *Format* yalnızca sekiz karakter uzunluğunda olduğu için, nesnenin adı en çok sekiz boşluk karakteri olmayan karakterlere sahip olmalıdır.

   A .DEF file called amqsvfcn.def is also supplied in the samples directory, *MQ_INSTALLATION_PATH*\Tools\C\Samples. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir. Bu dosyanın bir kopyasını alın ve yeniden adlandırın; örneğin, MYFORMAT.DEF' a. Yaratılmakta olan DLL adının ve MYFORMAT.DEF değeri aynıdır. Overwrite the name FORMAT1 in MYFORMAT.DEF with the new format name.

   Bir iletiyi her gönderdiğinizde bu adı kullanmayı unutmayın.
2. İletinizi göstermek için bir yapı oluşturun. Bir örnek için Geçerli sözdizimi konusuna bakın.
3. Veri dönüştürme çıkışınız için bir kod parçası oluşturmak üzere bu yapıyı `crtmqcvx` komutu aracılığıyla çalıştırın.

   CRTMQCVX komutu tarafından üretilen işlevler, tüm yapıların paketlenmiş olduğu varsayılarak yazılmış makroları kullanır; bu durumda, bu durumda durum bu değilse, bunları sona erdirir.
4. Copy the supplied skeleton source file, amqsvfc0.c, renaming it to the name of your message format that you set in step “1” sayfa 402.

   amqsvfc0.c , *MQ_INSTALLATION_PATH*\Tools\C\Samples dizininde; burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizindir. (Varsayılan kuruluş dizini: C:\Program Files\IBM\WebSphere MQ.)

   The skeleton includes a sample header file amqsvmha.h in the *MQ_INSTALLATION_PATH*\Tools\C\include directory. Bu dosyayı almak için, içerme yolunuzun bu dizini gösterdiğinden emin olun.

   amqsvmha.h dosyası, CRTMQCVX komutu tarafından oluşturulan kod tarafından kullanılan makroları içerir. Dönüştürülecek yapı karakter verisi içeriyorsa, bu makrolar MQXCNVC ' yi çağırır.
5. Kaynak dosyada aşağıdaki açıklama kutularını bulun ve anlatıldığı gibi kodu ekleyin:

a. Kaynak dosyanın sonuna doğru, bir yorum kutusu şununla başlar:

```
/* Insert the functions produced by the data-conversion exit */
```

Burada, “3” sayfa 402adımında oluşturulan kod parçasını ekleyin.

b. Kaynak dosyanın ortasındaki bir yorum kutusu şununla başlar:

```
/* Insert calls to the code fragments to convert the format's */
```

Bunu, `Converttag`STRUCTişlevine ilişkin açıklama satırı yapılan bir çağrıyı izlemektedir.

İşleve ilişkin adı, “5.a” sayfa 403adımında eklediğiniz işlevin adına çevirin. İşlevi etkinleştirmek için açıklama karakterlerini kaldırın. Birden çok işlev varsa, bunların her biri için çağrı yaratın.

c. Kaynak dosyanın başlangıcındaki bir açıklama kutusu şununla başlar:

```
/* Insert the function prototypes for the functions produced by */
```

Here, insert the function prototype statements for the functions added in step “3” sayfa 402.

6. Aşağıdaki komut dosyasını oluşturun:

```
cl -I MQ_INSTALLATION_PATH\Tools\C\Include -Tp   \
MYFORMAT.C

MYFORMAT.DEF
```

Burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizindir.

7. Çıkışınızı bir DLL dosyası olarak derlemek için komut kütüğünü verin.
8. Çıkışı, çıkış alt dizinine WebSphere MQ veri dizininin altına yerleştirin. Çıkışlarınızı 32 bit sistemlerle kurmak için kullanılan varsayılan dizin *MQ_DATA_PATH*\`Exits` ve 64 bit sistemler için *MQ_DATA_PATH*\`Exits64`dizinidir.

Veri dönüştürme çıkışlarını aramak için kullanılan yol kayıtta verilmiştir. Kayıt dosyası klasörü:

```
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\MQSeries\CurrentVersion\Configuration\ClientExitPa
th\
```

ve kayıt defteri anahtarı: `ExitsDefaultPath`. Bu yol, her bir kuyruk yöneticisi için ayarlanabilir ve çıkış yalnızca o yol ya da yollardaki aralara ararlıdır.

Not:

1. CRTMQCVX paketlenmiş yapıları kullanıyorsa, tüm WebSphere MQ uygulamaları bu şekilde derlenmelidir.
2. Veri dönüştürme çıkış programları yeniden girişli olmalıdır.
3. MQXCNVC, bir veri dönüştürme işleminden çıkarılabilen *yalnızca* MQI çağrısıdır.

Pencereler işletim sistemlerinde yükleme dosyalarını açma ve değiştirme

IBM WebSphere MQ for Windows Version 7.5 kuyruk yöneticisi işlemleri 32 bit 'tür. Sonuç olarak, 64 bit kullanan uygulamalar kullanılırken, bazı çıkış tipleri ve XA anahtar yükleme dosyalarının kuyruk yöneticisi tarafından kullanılmak üzere 32 bit sürümünün kullanılabilir olması gerekir. Çıkış ya da XA anahtarı yükleme dosyasının 32 bit sürümü gerekliyse ve kullanılamıyorsa, ilgili API çağrısı ya da komutu başarısız olur.

Two attributes are supported in the `qm.ini file`
for *ExitPath*. Bunlar `ExitsDefaultPath=`*MQ_INSTALLATION_PATH*`\exits` ve `ExitsDefaultPath64=`*MQ_INSTALLATION_PATH*`\exits64`' dir. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder. Bunlar, uygun kitaplığın bulunabilmesini

sağlar. Bir WebSphere MQ kümesinde bir çıkış kullanılırsa, bu durum uzak bir sistemde uygun kitaplığın bulunabilmesini sağlar.

Aşağıdaki çizelgede, 32 bit ya da 64 bit kullanan uygulamaların kullanılmakta olup olmadığına göre, farklı çıkış tipleri ve anahtar yükleme dosyaları ve 32 bit ya da 64 bit sürümlerinin ya da her ikisinin de gerekli olup olmadığını listelemektedir:

Dosya türleri	**32 bit uygulamalar**	**64 bit uygulamalar**
API geçiş çıkışı	32 bit	32 bit ve 64 bit
Veri dönüştürme çıkışı	32 bit	64 bit
Sunucu Kanalı çıkışları (tüm tipler)	32 bit	32 bit
İstemci Kanalı çıkışları (tüm tipler)	32 bit	64 bit
Kurulabilir hizmet çıkışı	32 bit	32 bit
Hizmet izleme birimi	32 bit	32 bit ve 64 bit
Küme WLM çıkışı	32 bit	32 bit
Pub/Alt yönlendirme çıkışı	32 bit	32 bit
Veritabanı anahtarı yükleme dosyaları	32 bit	32 bit ve 64 bit
Dış İşlem Yöneticisi AX kitaplıkları	32 bit	64 bit

Bir havuzdaki bağlanma öncesi çıkışı kullanarak bağlantı tanımlarına gönderme yapılması

WebSphere MQ MQI istemcileri, bağlantı öncesi bir çıkış kitaplığını kullanarak bağlantı tanımlarını elde etmek için bir havuz aramak üzere yapılandırılabilir.

Giriş

İstemci uygulaması, istemci kanal tanımlama çizelgelerini (CCDT) kullanarak bir kuyruk yöneticisine bağlanabiliyor. Genel olarak, CCDT dosyası merkezi bir ağ dosya sunucusunda bulunur ve bu dosyaya gönderme yapan istemcilere sahiptir. CCDT dosyasına gönderme yapan çeşitli istemci uygulamalarının yönetilmesi ve yönetilmesi zor olduğundan, esnek bir yaklaşım, istemci tanımlamalarının bir LDAP dizini, bir WebSphere Registry and Repository ya da başka bir havuz gibi genel bir havuzda saklamasıdır. Bir havuzda istemci bağlantısı tanımlarının saklanması, istemci bağlantısı tanımlarının yönetilmesini kolaylaştırır ve uygulamalar, doğru ve en güncel istemci bağlantısı tanımlamalarına erişebilir.

MQCONN/X çağrısı yürütülürken, IBM WebSphere MQ MQI client , bir uygulamanın önceden bağlanma öncesi çıkış kitaplığını yükler ve bağlantı tanımlamalarını almak için bir çıkış işlevini çağırır. Daha sonra, alınan bağlantı tanımları bir kuyruk yöneticiyle bağlantı kurmak için kullanılır. Çağrılacak çıkış kitaplığı ve işlevinin ayrıntıları mqclient.ini yapılandırma dosyasında belirtilir.

Sözdizimi

`void MQ_PRECONNECT_EXIT` (*pExitParms*, *pQMgrAd*, *ppConnectOpts*, *pCompKodu*, *pReason*);

Parametreler

pExitParms

Tip: PMQNXP giriş /çıkış

`PreConnection` çıkış değiştirgesi yapısı.

Yapı, çıkışa ilişkin çağırıcı tarafından ayrılır ve sürdürür.

pQMgrAdı
Tip: PMQCHAR giriş/çıkış

Kuyruk yöneticisinin adı.

On input, this parameter is the filter string supplied to the MQCONN API call through the **QMgrName** parameter. Bu alan boş, açık ya da belirli genel arama karakterleri içerebilir. Alan, çıkışa göre değiştirilir. Çıkış MQXR_TERM ile çağrıldığında, parametre NULL (boş değerli) olur.

ppConnectSeçenekleri
Tip: ppConnectGiriş/çıkış

MQCONNX 'in işlemini denetleyen seçenekler.

Bu, MQCONN API çağrısının işlemini denetleyen MQCNO bağlantı seçenekleri yapısına bir işarettir. Çıkış MQXR_TERM ile çağrıldığında, parametre NULL (boş değerli) olur. MQI istemcisi, çıkışa her zaman, uygulama tarafından sağlanmamış olsa da, çıkışa bir MQCNO yapısı sağlar. Bir uygulama MQCNO yapısı sağlıyorsa, istemci bunu değiştirdiği yerden çıkışa geçirmek için bir yineleme yapar. İstemci MQCNO ' nun sahipliğini korur.

MQCNO ile gönderme yapılan bir MQCD, dizi aracılığıyla sağlanan bağlantı tanımlarına göre öncelik kazanır. İstemci, kuyruk yöneticisine bağlanmak için MQCNO yapısını kullanıyor ve diğerleri yok sayılıyor.

pCompKodu
Tip: PMQXX_ENCODE_CASE_ONE long giriş/çıkış

Tamamlanma kodu.

Çıkışların tamamlanma kodunu alan bir MQUZE işaretçisi. Bu değer aşağıdaki değerlerden biri olmalıdır:

- `MQCC_OK` -Tamamlama tamamlandı
- `MQCC_UYARI` -Uyarı (kısmi tamamlama)
- `MQCC_FAILED` -Arama başarısız oldu

pReason
Tip: PMQXX_ENCODE_CASE_ONE long giriş/çıkış

Neden niteleyici pCompkodu.

Çıkış neden kodunu alan bir MQUZE işaretçisi. Tamamlanma kodu MQCC_OK ise, tek geçerli değer şöyledir:

- MQRC_NONE-(0, x '000') Raporlamak için bir neden yok.

Tamamlanma kodu MQCC_FAILED ya da MQCC_UYARI ise, çıkış işlevi neden kodu alanını geçerli bir MQRC_ * değerine ayarlayabiliyor.

C Çağırma

```
void MQ_PRECONNECT_EXIT (&ExitParms, &QMgrName, &pConnectOpts, &CompCode, &Reason);
```

Parameter

```
PMQNXP  pExitParms   /*PreConnect exit parameter structure*/
PMQCHAR pQMgrName    /*Name of the queue manager*/
PPMQCNO ppConnectOpts/*Options controlling the action of MQCONNX*/
PMQLONG pCompCode    /*Completion code*/
PMQLONG pReason      /*Reason qualifying pCompCode*/
```

İstemci yapılandırma dosyasınınPreConnect kısmı

`mqclient.ini` dosyasındaki PreConnect çıkışını yapılandırmak için PreConnect kısmını kullanın.

PreConnect kısmına aşağıdaki öznitelikler eklenebilir:

Data=*< URL>*
Bağlantı tanımlarının depolandığı havuzun URL adresi. Örneğin, bir LDAP sunucusu kullanılırken:

Veri = ldap://myLDAPServer.com:389/cn=wmq,ou=ibm,ou=com

Function=*<myFunc>*
İşlevsel giriş noktasının adı, PreConnect çıkış kodunu içeren kitaplığa ilişkin ad.

İşlev tanımlaması PreConnect çıkış prototipine (MQ_PRECONNECT_EXIT) bağlı olur.

Bu alanın uzunluk üst sınırı MQ_EXIT_NAME_LENGTH ' dir.

Module=*< amqldapi>*
API çıkış kodunu içeren modülün adı.

Bu alan modülün tam yol adını içeriyorsa, olduğu gibi kullanılır.

Sequence=*< sequence_number>*
Bu çıkışa diğer çıkışlara göre çağrıldığı sıra. Sıra numarası düşük olan bir çıkış, daha yüksek sıra numarasına sahip bir çıkıştan önce çağrılır. Çıkışların sıra numaralandırmasına sürekli olarak gerek yoktur; 1, 2, 3 sırası, 7, 42, 1096 sırası ile aynı sonucu elde eder. Bu öznitelik işaretsiz bir sayısal değer.

`mqclient.ini` dosyası içinde birden çok PreConnect kısmı tanımlanabilir. Her çıkışa ilişkin işleme sırası, stanza 'nın Sıra özniteliği tarafından belirlenir.

Yayınlama çıkışlarının yazılması ve derlenmesi

Yayınlanan bir iletinin içeriğini aboneler tarafından alınmadan önce değiştirmek için, kuyruk yöneticisinde bir yayınlama çıkışı yapılandırabilirsiniz. Ayrıca, ileti üstbilgisini değiştirebilir ya da iletiyi bir aboneliğe teslim etmemeniz de olabilir.

z/OSüzerinde yayınlama çıkışları desteklenmez.

Abonelere teslim edilen iletileri incelemek ve değiştirmek için yayınlama çıkışını kullanabilirsiniz:

- Her abonede yayınlanan bir iletinin içeriğini inceler
- Her abonede yayınlanan bir iletinin içeriğini değiştirme
- İletinin konacağı kuyruğu değiştir
- Bir iletiyi aboneye teslim etmeyi durdurur

Yayınlama çıkışı yazılıyor

Çıkışınızı yazmanıza ve derlemenize yardımcı olması için “Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357içindeki adımları kullanın.

Yayınlama çıkışa ilişkin sağlayıcı, çıkışa ilişkin bilgileri tanımlar. Ancak çıkış, MQPSXP' de tanımlanan kurallara uygun olmalıdır.

WebSphere MQ , MQ_PUBLISH_EXIT giriş noktasının bir somutlamasını sağlamıyor. Bu, bir C dili tipidef bildirimi sağlar. Değiştirgelerin kullanıcı tarafından yazılmış bir çıkışa doğru olarak bildirilmesi için typedef 'i kullanın. Aşağıdaki örnekte, typedef bildiriminin nasıl kullanılacağı gösterilmektedir:

```
#include "cmqec.h"

MQ_PUBLISH_EXIT MyPublishExit;

void MQENTRY MyPublishExit( PMQPSXP pExitParms,
                            PMQPBC  pPubContext,
                            PMQSBC  pSubContext )
{
  /* C language statements to perform the function of the exit */
}
```

Aşağıdaki işlemlerin bir sonucu olarak, yayınlama çıkışı kuyruk yöneticisi işlemi içinde çalışır:

- Bir iletinin bir ya da daha fazla aboneye teslim edildiği bir yayınlama işlemi
- Bir ya da daha fazla tutulan iletinin teslim edildiği bir abone olma işlemi
- Bir ya da daha fazla tutulan iletinin teslim edildiği bir Abonelik İsteği işlemi

Yayınlama çıkışı bir bağlantı için çağrılırsa, ilk olarak *ExitReason* kodu MQXR_INIT olarak adlandırılır. Before the connection disconnects after using a publish exit, the exit is called with an *ExitReason* code of MQXR_TERM.

Yayınlama çıkışı yapılandırıldıysa, ancak kuyruk yöneticisi başlatıldığında, ancak kuyruk yöneticisi için yayınlama/abone olma ileti işlemleri engellendiğinde yüklenemez. Yayınlama/abone olma ileti sistemi yeniden etkinleştirilmeden önce sorunu düzeltmeniz ya da kuyruk yöneticisini yeniden başlatmanız gerekir.

Yayınlama çıkışını gerektiren her WebSphere MQ bağlantısı, çıkışı yükleyemeyebilir ya da kullanıma hazırlamayabilir. Çıkış yükleme ya da kullanıma hazırlama işlemi başarısız olursa, yayınlama çıkışını gerektiren abone olma/abone olma işlemleri ilgili bağlantı için geçersiz kılınır. İşlemler, WebSphere MQ neden kodu MQRC_PUBLISH_EXIT_ERRORile başarısız olur.

Yayınlama çıkışımın çağrıldığı bağlam, bir uygulama tarafından kuyruk yöneticisine yönelik bağlantıdır. Bir kullanıcı veri alanı, yayınlama işlemleri gerçekleştiren her bir bağlantı için kuyruk yöneticisi tarafından korunur. Çıkış, her bağlantı için kullanıcı verileri alanındaki bilgileri saklayabilir.

Yayınlama çıkışı bazı MQI çağrılarını kullanabilir. Bu, yalnızca ileti özelliklerini yönlendiren bu MQI çağrılarını kullanabilir. Aramalar:

- MQBUFMH
- MQCRTMH
- MQDLTMH
- MQDLTMP
- MQMHBUF
- MQINQMP
- MQSETMP

Yayınlama çıkışı hedef kuyruk yöneticisini ya da kuyruk adını değiştirirse, yeni bir yetki denetimi gerçekleştirilmez.

Yayınlama çıkışısının derlenmesi

Yayınlama çıkışı dinamik olarak yüklenmiş bir kitaptır; kanal çıkışı olarak düşünülebilmektedir. Çıkışların derlenmesiyle ilgili bilgi için bkz. “Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357.

Örnek yayınlama çıkışı

Örnek çıkış programı amqspse0.colarak adlandırılır. Çıkış, başlatma, yayınlama ya da sonlandırma işlemleri için çıkışa çağrılıp çağrılmadığına bağlı olarak, günlük dosyasına farklı bir ileti yazar. Ayrıca, çıkış kullanıcı alanı alanının, depolamayı uygun bir şekilde ayırmak ve serbest bir şekilde serbest yapmak için kullanılması da gösterilir.

Yayınlama çıkışlarının yapılandırılması

Yayınlama çıkışı yapılandırmak için bazı öznitelikleri tanımlamanız gerekir.

Windows ve Linux üzerinde, öznitelikleri tanımlamak için WebSphere MQ gezginini kullanabilirsiniz. Öznitelikler, Publish/Subscribe altında, kuyruk yöneticisi özellikleri sayfasında tanımlanır.

UNIX ve Linux sistemlerinde qm.ini dosyasında yayınlama çıkışını yapılandırmak için PublishSubscribeadlı bir stanza oluşturun. PublishSubscribe Stanza aşağıdaki özniteliklere sahiptir:

PublishExitPath=[path]|*module_name*
Yayınlama çıkış kodunu içeren modül adı ve yolu. Bu alanın uzunluk üst sınırı MQ_EXIT_NAME_LENGTH' dir. Varsayılan değer yayınlama çıkışıdır.

PublishExitFunction=*function_name*
İşlev girdisi noktasının adı, yayınlama çıkış kodunu içeren modüle işaret eder. Bu alanın uzunluk üst sınırı MQ_EXIT_NAME_LENGTH' dir.

PublishExitData=*string*
Kuyruk yöneticisi bir yayınlama çıkışı çağırıyorsa, giriş olarak bir MQPSXP yapısını geçirir. The data specified using the *PublishExitData* attribute is provided in the *ExitData* field of the structure. Dize, MQ_EXIT_DATA_LENGTH karakterlerine kadar uzunluğa kadar çıkabilmektedir. Varsayılan değer 32 boş karakterdir.

Küme iş yükü çıkışlarının yazılması ve derlenmesi

Kümelerin iş yükü yönetimini özelleştirmek için bir küme iş yükü çıkış programı yazın. İletilerin yönlendirilmesi sırasında, günün farklı zamanlarında bir kanalı ya da ileti içeriğini kullanarak bir kanalı kullanma maliyetini de kullanabilirsiniz. Bunlar, standart iş yükü yönetimi algoritmasıyla dikkate alınmayan etkenlerdir.

Çoğu durumda, iş yükü yönetimi algoritması gereksinimleriniz için yeterli olur. Ancak, iş yükü yönetimini uyarlamak için kendi kullanıcı çıkışı programınızı sağlayabilmeniz için, WebSphere MQ bir kullanıcı çıkışı, küme iş yükü çıkışı içerir.

Ağınızla ya da iş yükü dengelemesini etkilemek için kullanabileceğiniz iletilerinizle ilgili bazı bilgilere sahip olabilirsiniz. Yüksek kapasiteli kanallar ya da ucuz ağ rotaları hangileridir, ya da iletileri içeriğine bağlı olarak yönlendirmek isteyebilirsiniz. Bir küme iş yükü çıkış programı yazmaya karar verebilir ya da bir üçüncü kişi tarafından sağlanan bir program kullanabilirsiniz.

Küme iş yükü çıkışı bir küme kuyruğuna erişilirken çağrılır. It is called by MQOPEN, MQPUT1 and MQPUT.

The target queue manager selected at MQOPEN time is fixed if MQOO_BIND_ON_OPEN is specified. Bu durumda, çıkış yalnızca bir kez çalıştırılır.

If the target queue manager is not fixed at MQOPEN time, the target queue manager is chosen at the time of the MQPUT call. Hedef kuyruk yöneticisi kullanılabilir değilse ya da ileti hala iletim kuyruğunda olduğunda başarısız olursa, çıkış yeniden çağrılır. Yeni bir hedef kuyruk yöneticisi seçildi. İleti aktarılırken ileti kanalı başarısız olursa ve ileti geriletilirse, yeni bir hedef kuyruk yöneticisi seçilir.

z/OSdışındaki platformlarda, kuyruk yöneticisi yeni küme iş yükü çıkışını kuyruk yöneticisinin bir sonraki başlatışımından yükler.

Kuyruk yöneticisi tanımlaması küme iş yükü çıkış programı adı içermiyorsa, küme iş yükü çıkışı çağrılmaz.

Çıkış parametresi yapısındaki bir küme iş yükü çıkışa çeşitli veriler iletilir, MQWXP:

- İleti tanımlaması yapısı (MQMD).
- İleti uzunluğu parametresi.
- İletinin bir kopyası ya da iletinin bir parçası.

z/OS dışı altyapılarda, CLWLMode=FASTkullanıyorsanız, her işletim sistemi işlemi çıkışa ait kendi kopyasını yükler. Kuyruk yöneticisine yapılan farklı bağlantılar, çıkışa ilişkin farklı kopyaların çağrılmasına neden olabilir. Çıkış varsayılan güvenli kipte çalıştırılırsa, CLWLMode=SAFE, çıkışa ait tek bir kopya kendi ayrı işleminde çalışır.

Küme iş yükü çıkışları yazılıyor

z/OSdışındaki platformlarda, küme iş yükü çıkışları için MQI çağrıları kullanılmamalıdır. diğer açıdan, küme iş yükü çıkış programlarını yazma ve derlemeye yönelik kurallar, kanal çıkış programları için geçerli olan kurallar gibidir. Follow the steps in “Çıkışların ve kurulabilen hizmetlerin yazılması ve derlenmesi” sayfa 357, and use the sample program, “Örnek küme iş yükü çıkışı” sayfa 409 to help write and compile your exit.

Kanal çıkışlarına ilişkin daha fazla bilgi için bkz. “Kanal-çıkış programları yazılıyor” sayfa 381.

Küme iş yükü çıkışlarının yapılandırılması

You name cluster workload exits in the queue manager definition by specifying the cluster workload exit attribute on the ALTER QMGR command. Örneğin:

```
ALTER QMGR CLWLEXIT(myexit)
```

Örnek küme iş yükü çıkışı

WebSphere MQ örnek bir küme iş yükü çıkış programı içerir. Örneği kopyalayabilir ve kendi programlarınız için temel olarak kullanabilirsiniz.

z/OSdışındaki platformlarda

Örnek küme iş yükü çıkış programı C ' de sağlanır ve adı `amqswlm0.` colarak adlandırılır. Bu öğe aşağıdaki yerde bulunabilir:

Çizelge 57. Örnek küme iş yükü çıkış programı yeri (z/OSdeğil)	
Altyapı	**filePath**
AIX, HP-UX, Sun Solaris	*MQ_INSTALLATION_PATH*/samp
Pencereler	*MQ_INSTALLATION_PATH*\Tools\c\Samples

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Bu örnek çıkış, kuyruk yöneticisi kullanılamaz duruma gelmediği sürece, tüm iletileri belirli bir kuyruk yöneticisine yönlendirir. Bu işlem, iletileri başka bir kuyruk yöneticisine yönelterek kuyruk yöneticisinin başarısızlığa uğramasını sağlar.

İletilerin hangi kuyruk yöneticisini göndermesini istediğinizi belirtin. Kuyruk yöneticisi tanımlamasındaki CLWLDATA özniteindeki küme alıcısı kanalının adını belirtin. Örneğin:

```
ALTER QMGR CLWLDATA('my-cluster-name.my-queue-manager')
```

Çıkışı etkinleştirmek için, CLAXLEXIT özniteisinde tam yolunu ve adını belirtin:

UNIX and Linux sistemlerinde:

```
ALTER QMGR CLWLEXIT('path/amqswlm(clwlFunction)')
```

Windows sistemlerinde:

```
ALTER QMGR CLWLEXIT('path\amqswlm(clwlFunction)')
```

Şimdi, sağlanan iş yükü yönetimi algoritmasını kullanmak yerine, WebSphere MQ bu çıkışı, tüm iletileri seçtiğiniz kuyruk yöneticinize yöneltmek için çağırır.

IBM WebSphere MQ uygulaması oluşturulması

Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

Uygulamanızı AIXüzerinde oluşturma

AIX yayınları, yazdığınız programlardan yürütülebilir uygulamaların nasıl oluşturulacağını açıklar.

This topic describes the additional tasks, and the changes to the standard tasks, that you must perform when building WebSphere MQ for AIX applications to run under AIX. C, C + + ve COBOL desteklenmektedir. C++ programlarınızı hazırlamaya ilişkin bilgi için bkz. C++ kullanılması.

AIX için WebSphere MQ kullanarak yürütülebilir bir uygulama oluşturmak için gerçekleştirmeniz gereken görevler, kaynak kodunuzun yazıldığı programlama diline göre değişiklik gösterir. In addition to coding the MQI calls in your source code, you must add the appropriate language statements to include the WebSphere MQ for AIX include files for the language that you are using. Bu dosyaların içeriğini kendinize tanıdık bir hale getiriniz. Tam açıklama için bkz. “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77 .

İş parçacıklı sunucu ya da iş parçacıklı istemci uygulamaları çalıştırdığınızda, AIXTHREAD_SCOPE = S. ortam değişkenini ayarlayın.

C programlarının AIXiçinde hazırlanması

This topic contains information about linking libraries necessary to prepare C programs on AIX.

Önderlenmiş C programları, *MQ_INSTALLATION_PATH*/samp/bin dizininde sağlanır. ANSI derleyicisini kullanın ve aşağıdaki komutları çalıştırın. 64 bit kullanan uygulamalar hakkında daha fazla bilgi için bkz. 64 bit altyapılarda koşma standartları.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

32 bit uygulamalar için:

```
  $ xlc_r -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -lmqm
```

Burada `amqsput0` , örnek bir programdır.

64 bit uygulamalar için:

```
  $ xlc_r -q64 -o amqsput_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64
-lmqm
```

Burada `amqsput0` , örnek bir programdır.

C++ programları için VisualAge C/C++ derleyicisini kullanıyorsanız, kitaplıkları bağlarken çözümlenen tüm WebSphere MQ simgelerini almak için `-q namemangling=v5` seçeneğini eklemeniz gerekir.

If you want to use the programs on a machine that has only the WebSphere MQ MQI client for AIX installed, recompile the programs to link them with the client library (`-lmqic`) instead.

Kitaplıkların bağlanması

Aşağıdaki kitaplıklara gereksinim duyarsınız:

- Programlarınızı, WebSphere MQtarafından sağlanan uygun kitaplıkla bağlantılayın.

  İş parçacıklı olmayan bir ortamda, aşağıdaki kitaplıklardan birine bağlayın:

Kitaplık dosyası	Program/çıkış tipi
libmqm.a	C sunucusu için sunucu
libmqic.a & libmqm.a	C İçin İstemci

  İş parçacıklı bir ortamda, aşağıdaki kitaplıklardan birine bağlantı:

Kitaplık dosyası	Program/çıkış tipi
libmqm_r.a	C sunucusu için sunucu
libmqic_r.a & libmqm_r.a	C İçin İstemci

  Örneğin, tek bir derleme biriminden basit bir iş parçacıklı WebSphere MQ uygulaması oluşturmak için aşağıdaki komutları çalıştırın.

  32 bit uygulamalar için:

```
$ xlc_r -o amqsputc_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -lmqm_r
```

Burada `amqsput0` , örnek bir programdır.

64 bit uygulamalar için:

```
$ xlc_r -q64 -o amqsputc_64_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64
-lmqm_r
```

Burada `amqsput0` , örnek bir programdır.

If you want to use the programs on a machine that has only the WebSphere MQ MQI client for AIX installed, recompile the programs to link them with the client library (`-lmqic`) instead.

Not:

1. Kurulabilir bir hizmet yazıyorsanız (ek bilgi için bkz. Yönetme), iş parçacıklı olmayan bir uygulamadaki `libmqmzf.a` kitaplığına ve iş parçacıklı bir uygulamadaki `libmqmzf_r.a` kitaplığına bağlanmanız gerekir.
2. If you are producing an application for external coordination by an XA-compliant transaction manager such as IBM TXSeries, Encina, or BEA Tuxedo, you need to link to the `libmqmxa.a` (or `libmqmxa64.a` if your transaction manager treats the 'long' type as 64 bit) and `libmqz.a` libraries in a non-threaded application and to the `libmqmxa_r.a` (or `libmqmxa64_r.a`) and `libmqz_r.a` libraries in a threaded application.
3. Güvenilen uygulamaları iş parçacıklı WebSphere MQ kitaplıklarına bağlamaya gerek duyarsınız. However, only one thread in a trusted application on WebSphere MQ on UNIX and Linux systems can be connected at a time.
4. Diğer ürün kitaplıklarından önce WebSphere MQ kitaplıklarını bağlamanız gerekir.

Preparing COBOL programs in AIX

Use this information when preparing COBOL programs in AIX using IBM COBOL Set and Micro Focus COBOL.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

- 32 bit COBOL kopya kitapları aşağıdaki dizine takılır:

  ```
  MQ_INSTALLATION_PATH/inc/cobcpy32
  ```

  ve simgesel bağlantılar aşağıdaki yerde yaratılır:

  ```
  MQ_INSTALLATION_PATH/inc
  ```

- 64 bit COBOL kopya kitapları aşağıdaki dizine takılır:

  ```
  MQ_INSTALLATION_PATH/inc/cobcpy64
  ```

Aşağıdaki örneklerde, **COBCPY** ortam değişkeni şu şekilde ayarlanır:

```
MQ_INSTALLATION_PATH/inc/cobcpy32
```

32 bit uygulamalar için ve:

```
MQ_INSTALLATION_PATH/inc/cobcpy64
```

64 bit uygulamalar için.

Programınızı aşağıdaki kitaplık dosyalarından biriyle bağlamaya gereksinim duyarsınız:

Kitaplık dosyası	Program/çıkış tipi
libmqmcb.a	COBOL için sunucu (iş parçacıklı uygulama)
libmqmcb_r.a	COBOL için sunucu (iş parçacıklı uygulama)
libmqicb.a	COBOL için İstemci (iş parçacıklı uygulama
libmqicb_r.a	COBOL için İstemci (iş parçacıklı uygulama)

You can use the IBM COBOL Set compiler or Micro Focus COBOL compiler depending on the program:

- amqm ' un başlangıç programları Micro Focus COBOL derleyicisi için uygundur ve
- amq0 başlangıç programları, her iki derleyici için uygundur.

Preparing COBOL programs using IBM COBOL Set for AIX

Örnek COBOL programları IBM WebSphere MQile sağlanır. Böyle bir programı derlemek için, aşağıdaki listeden uygun komutu girin:

32 bit yivli olmayan sunucu uygulaması

```
$ cob2 -o amq0put0 amq0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb -qLIB \
-I<COBCPY>
```

32 bit yivli olmayan istemci uygulaması

```
$ cob2 -o amq0put0 amq0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb -qLIB \
-I<COBCPY>
```

32 bit yivli sunucu uygulaması

```
$ cob2_r -o amq0put0 amq0put0.cbl -qTHREAD -L MQ_INSTALLATION_PATH/lib \
-lmqmcb_r -qLIB -I<COBCPY>
```

32 bit yivli istemci uygulaması

```
$ cob2_r -o amq0put0 amq0put0.cbl -qTHREAD -L MQ_INSTALLATION_PATH/lib \
-lmqicb_r -qLIB -I<COBCPY>
```

64 bit yivli olmayan sunucu uygulaması

```
$ cob2 -o amq0put0 amq0put0.cbl -q64 -L MQ_INSTALLATION_PATH/lib - lmqmcb \
-qLIB -I<COBCPY>
```

64 bit yivli olmayan istemci uygulaması

```
$ cob2 -o amq0put0 amq0put0.cbl -q64 -L MQ_INSTALLATION_PATH/lib - lmqicb \
-qLIB -I<COBCPY>
```

64 bit yivli sunucu uygulaması

```
$ cob2_r -o amq0put0 amq0put0.cbl -q64 -qTHREAD -L MQ_INSTALLATION_PATH/lib \
-lmqmcb_r -qLIB -I<COBCPY>
```

64 bit yivli istemci uygulaması

```
$ cob2_r -o amq0put0 amq0put0.cbl -q64 -qTHREAD -L MQ_INSTALLATION_PATH/lib \
-lmqicb_r -qLIB -I<COBCPY>
```

COBOL programlarının Micro Focus COBOL

Programınızı derlemeden önce ortam değişkenlerini aşağıdaki gibi ayarlayın:

```
export COBCPY=<COBCPY>
export LIBPATH=MQ_INSTALLATION_PATH/lib:$LIBPATH
```

Bir 32 bit COBOL programını Micro Focus COBOL kullanarak derlemek için şunu girin:

- COBOL için sunucu

```
$ cob32 -xvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb
```

- COBOL için İstemci

```
$ cob32 -xvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb
```

- COBOL İçin İş Y

```
$ cob32 -xtvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb_r
```

- COBOL için İş parçacığı

```
$ cob32 -xtvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb_r
```

Bir 64 bit COBOL programını Micro Focus COBOL kullanarak derlemek için şunu girin:

- COBOL için sunucu

```
$ cob64 -xvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb
```

- COBOL için İstemci

```
$ cob64 -xvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb
```

- COBOL İçin İş Y

```
$ cob64 -xtvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb_r
```

- COBOL için İş parçacığı

```
$ cob64 -xtvP amqminqx.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb_r
```

Burada amqminqx , örnek bir programdır

Ayarlamanız gereken ortam değişkenlerine ilişkin açıklamalar için Micro Focus COBOL belgelerine bakın.

Preparing CICS application programs in AIX

Use this information when preparing CICS programs in AIX.

XA anahtar modülleri, CICS ile IBM WebSphere MQ arasında bağlantı sağlamanıza olanak sağlamak için sağlanmıştır:

Çizelge 58. AIXüzerinde CICS uygulama programları için temel kod: XA kullanıma hazırlama yordamı

Tanım	**C (kaynak)**	**C (exec)-sitenize ekle XAD.Stanza**
XA kullanıma hazırlama yordamı	amqzscix.c	*amqzsc* - AIXiçin CICS

Use the prebuilt version of the IBM WebSphere MQ switch load file *amqzsc* , which is provided with the product.

C işlemlerinizi her zaman iş parçacığı korumalı IBM WebSphere MQ kitaplığınızla (*libmqm_r.a) bağlayın.* and your COBOL transactions with the COBOL library *libmqmcb_r.a.*.

You can find more information about supporting CICS transactions in the Yönetme.

TXSeries CICS desteği

AIX üzerindeIBM WebSphere MQ , XA arabirimini kullanarak TXSeries CICS ' i destekler. CICS uygulamalarının, IBM WebSphere MQ kitaplıklarının iş parçacıklı sürümüyle bağlantılı olduğundan emin olun.

CICS programlarını, IBM COBOL Set for AIX ya da Micro Focus COBOL kullanarak çalıştırabilirsiniz. Aşağıdaki kısımlarda, IBM COBOL Set for AIX ve Micro Focus COBOL üzerindeki CICS programlarının çalıştırılmasına ilişkin fark açıklanmaktadır.

C ya da COBOL ' de aynı CICS bölgesine yüklenen WebSphere MQ programları yazın. Aynı CICS bölgesine C ve COBOL MQI çağrılarının birleşiminden oluşan bir birleşim yapamazsınız. Most MQI calls in the second language used fail with a reason code of MQRC_HOBJ_ERROR.

Preparing CICS COBOL programs using IBM COBOL Set for AIX

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

IBM COBOL olanağını kullanmak için aşağıdaki adımları izleyin:

1. Aşağıdaki ortam değişkenini dışa aktarın:

```
export LDFLAGS="-qLIB -bI:/usr/lpp/cics/lib/cicsprIBMCOB.exp  \
                -IMQ_INSTALLATION_PATH/inc -I/usr/lpp/cics/include  \
                -e _iwz_cobol_main  \
```

   Burada LIB bir derleyici yönergesi 'dir.

2. Programı çevirerek, derleyin ve bağlayın:

```
cicstcl -l IBMCOB <yourprog>.ccp
```

Preparing CICS COBOL programs using Micro Focus COBOL

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

Micro Focus COBOL olanağını kullanmak için aşağıdaki adımları izleyin:

1. Aşağıdaki komutu kullanarak, IBM WebSphere MQ COBOL yürütme ortamı kitaplık modülünü yürütme ortamı kitaplığına ekleyin:

   ```
   cicsmkcobol -L/usr/lib/dce -LMQ_INSTALLATION_PATH/lib  \
               MQ_INSTALLATION_PATH/lib/libmqmcbrt.o -lmqe_r
   ```

   Not: `cicsmkcobol`ile, IBM WebSphere MQ , C programlama dilinde COBOL uygulamanızın MQI çağrılarını gerçekleştirmenize izin vermez.

   Var olan uygulamalarınızda bu tür çağrılar varsa, bu işlevleri COBOL uygulamalarından kendi kitaplığınıza taşımanız önerilir; örneğin, `myMQ.so`. After moving the functions, do not include the IBM WebSphere MQ library `libmqmcbrt.o` when building the COBOL application for CICS.

   Ayrıca, COBOL uygulamanızın COBOL MQI çağrısı yapmazsa, `libmqmz_r` ile `cicsmkcobol`bağlantısını bağlamayın.

   Bu, Micro Focus COBOL dil yöntemi dosyasını oluşturur ve CICS yürütme ortamı COBOL kitaplığını UNIX and Linux sistemlerinde IBM WebSphere MQ ' i çağırmak için etkinleştirir.

   Not: `cicsmkcobol` komutunu yalnızca aşağıdaki ürünlerden birini kurduğunuzda çalıştırın:

   - Micro Focus COBOL ' in yeni sürümü veya
   - Yeni sürüm ya da yayın düzeyi: CICS for AIX
   - Desteklenen herhangi bir veritabanı ürününün yeni sürümü ya da yayın düzeyi (yalnızca COBOL işlemleri için)
   - New version or release of IBM WebSphere MQ
2. Aşağıdaki ortam değişkenini dışa aktarın:

   ```
   COBCPY=MQ_INSTALLATION_PATH/inc  export COBCPY
   ```
3. Programı çevirerek, derleyin ve bağlayın:

   ```
   cicstcl -l COBOL -e <yourprog>.ccp
   ```

CICS C programlarının hazırlanması

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

Standart CICS olanaklarını kullanarak CICS C programlarını oluşturun:

1. Aşağıdaki ortam değişkenlerinin ***bir*** öğesini dışa aktarın:
   - LDflags = "-L/*MQ_INSTALLATION_PATH*lib -lmqm_r" LDFLAGS dışa aktar
   - USERLIB = "-L*MQ_INSTALLATION_PATH*lib -lmqm_r" export USERLIB
2. Programı çevirerek, derleyin ve bağlayın:

   ```
   cicstcl -l C amqscic0.ccs
   ```

CICS C örnek hareketi

Bir AIX IBM WebSphere MQ işlemi için Örnek C kaynağı, AMQSCIC0.CCS. Hareket, varsayılan kuyruk yöneticisinde SYSTEM.SAMPLE.CICS.WORKQUEUE iletim kuyruğundan iletileri okur ve bunları, iletinin iletim üstbilgisinde yer alan bir kuyruk adıyla yerel kuyruğa yerleştirir. SYSTEM.SAMPLE.CICS.DLQ.

Örnek MQSC komut kütüğünü (AMQSCIC0.TST , bu kuyrukları ve örnek giriş kuyruklarını yaratmanızı sağlar.

Uygulamanızı HP Integrity NonStop Serverüzerinde oluşturma

This information describes the additional tasks, and the changes to the standard tasks, that you must perform when you build IBM WebSphere MQ client for HP Integrity NonStop Server applications to run under HP Integrity NonStop Server.

C, COBOL ve pTAL desteklenir.

OSS ve Guardian üstbilgileri ve genel kitaplıklar

OSS ve Guardian üstbilgileri ve genel kitaplıkların listeleri sağlar. Listelenen OSS üstbilgileri, OSS genel yürütülür dosyası ve genel içe aktarma kitaplıkları, Guardian üstbilgileri ve Guardian genel yürütülebilir dosyaları ve genel içe aktarma kitaplıkları vardır.

“OSS üstbilgileri” sayfa 416

“OSS genel yürütülür dosyası ve genel içe aktarma kitaplıkları” sayfa 417

“Koruyucu üstbilgiler” sayfa 417

“Koruyucu genel yürütülür dosya ve genel içe aktarma kitaplıkları” sayfa 418

OSS üstbilgileri

Çizelge 59. OSS üstbilgileri

Nesne	Konum	Tanım
cmqbc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqfc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqec.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqpsc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqxc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqzc.h	<mqinstall>/inc	IBM WebSphere MQ C dili üstbilgisi (OSS)
cmqcobol.cpy	<mqinstall>/inc	IBM WebSphere MQ COBOL copybook (OSS)
cmqbt.tal	<mqinstall>/inc	IBM WebSphere MQ pTAL üstbilgisi (OSS)
cmqcft.tal	<mqinstall>/inc	IBM WebSphere MQ pTAL üstbilgisi (OSS)
cmqpst.tal	<mqinstall>/inc	IBM WebSphere MQ pTAL üstbilgisi (OSS)

Çizelge 59. OSS üstbilgileri (devamı var)		
Nesne	**Konum**	**Tanım**
cmqt.tal	<mqinstall>/inc	IBM WebSphere MQ pTAL üstbilgisi (OSS)
cmqxt.tal	<mqinstall>/inc	IBM WebSphere MQ pTAL üstbilgisi (OSS)

OSS genel yürütülür dosyası ve genel içe aktarma kitaplıkları

Çizelge 60. OSS genel yürütülür dosyası ve genel içe aktarma kitaplıkları		
Nesne	**Konum**	**Tanım**
libmqic.so	<mqinstall>/bin	IBM WebSphere MQ genel yürütülebilir kitaplığı (OSS iş parçacıklı)
libmqic_r.so	<mqinstall>/bin	IBM WebSphere MQ genel yürütülebilir kitaplığı (OSS çok iş parçacıklı)
libmqic.so	<mqinstall>/lib	IBM WebSphere MQ genel içe aktarma kitaplığı (OSS iş parçacıklı)
libmqic_r.so	<mqinstall>/lib	IBM WebSphere MQ genel içe aktarma kitaplığı (OSS çok iş parçacıklı)
mqicb	<mqinstall>/lib	COBOL (OSS) içinIBM WebSphere MQ genel içe aktarma kitaplığı

Koruyucu üstbilgiler

Çizelge 61. Koruyucu üstbilgiler		
Nesne	**Konum**	**Tanım**
cmqbch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqfch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqech	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqpsch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqxch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)
cmqzch	<mqinstall>/inc/G	IBM WebSphere MQ C-dil üstbilgisi (Guardian)

Çizelge 61. Koruyucu üstbilgiler (devamı var)		
Nesne	**Konum**	**Tanım**
cmqcobol	<mqinstall>/inc/G	IBM WebSphere MQ COBOL copybook (Koruyucu)
cmqbt	<mqinstall>/inc/G	IBM WebSphere MQ pTAL üstbilgisi (Guardian)
cmqcft	<mqinstall>/inc/G	IBM WebSphere MQ pTAL üstbilgisi (Guardian)
cmqpst	<mqinstall>/inc/G	IBM WebSphere MQ pTAL üstbilgisi (Guardian)
cmqt	<mqinstall>/inc/G	IBM WebSphere MQ pTAL üstbilgisi (Guardian)
cmqxt	<mqinstall>/inc/G	IBM WebSphere MQ pTAL üstbilgisi (Guardian)

Koruyucu genel yürütülür dosya ve genel içe aktarma kitaplıkları

Çizelge 62. Koruyucu genel yürütülür dosya ve genel içe aktarma kitaplıkları		
Nesne	**Konum**	**Tanım**
mqic	<mqinstall>/bin/G	IBM WebSphere MQ genel yürütülebilir kitaplığı (Guardian)
mqicb	<mqinstall>/lib/G	IBM WebSphere MQ public import library for COBOL (Guardian)

C programlarını HP Integrity NonStop Serveriçinde hazırlama

Bu konu, C programlarını, OSS C derleyicisini kullanırken ve Guardian C derleyicisini kullanırken, uygulamaları oluştururken kullandığınız komutların örnekleriyle birlikte HP Integrity NonStop Server ' ta bir araya getirdiğinizde dikkate alınacak bilgileri içerir.

Önderlenmiş C programları, MQ_INSTALLATION_PATH/opt/mqm/samp/bin dizininde sağlanır. Kaynak koddan bir örnek oluşturmak için, c89 derleyicisini kullanın.

Programlarınızı, IBM WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir. Aşağıdaki çizelge, C programlarını HP Integrity NonStop Serverüzerinde hazırlarken bağlantı gereken kitaplıkları listeler.

Çizelge 63. . HP Integrity NonStop Server bağlantı kitaplıkları	
Kitaplık	**Tanım**
libmqic.so	OSS iş parçacıklı
libmqic_r.so	OSS çok iş parçacıklı
mqic	Vasi

Çok iş parçacıklı yerel IBM WebSphere MQ uygulamaları, Posix User Threads (PUT) özelliğini kullanmalıdır. Bu üründeki Standard Posix Threads (SPT) desteği yoktur.

OSS C derleyicisini kullanarak uygulama oluşturulması

Bu bölümde, OSS derleyicisini kullanırken, OSS ya da Guardian için hedeflenen programlar oluşturmak için kullanılan komutlara ilişkin örnekler yer almaktadır.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

Aşağıdaki örnek, iş parçacıklı bir C istemcisi OSS uygulaması oluşturmasını sağlar:

```
c89 -Wsystype=oss -o amqsputc amqsput0.c -IMQ_INSTALLATION_PATH/opt/mqm/inc
-LMQ_INSTALLATION_PATH/opt/mqm/lib -lmqic
```

Aşağıdaki örnekte, çok iş parçacıklı bir C istemcisi OSS uygulaması oluşturulur:

```
c89 -Wsystype=oss -D_PUT_MODEL_ -o amqsputc amqsput0.c -IMQ_INSTALLATION_PATH/opt/mqm/inc
-LMQ_INSTALLATION_PATH/opt/mqm/lib -lmqic_r -lput
```

Aşağıdaki örnek, bir Guardian C istemci uygulamasını oluşturur:

```
c89 -Wsystype=guardian -o /G/vol/subvol/amqsputc amqsput0.c -IMQ_INSTALLATION_PATH/opt/mqm/inc
-LMQ_INSTALLATION_PATH/opt/mqm/lib/G -lmqic
```

Guardian C derleyicisini kullanarak uygulama oluşturulması

Bu bölümde, Guardian derleyicisini kullanırken Guardian için hedeflenen programları oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu Guardian birimini ve alt birimi temsil eder.

Aşağıdaki örnek, bir Guardian C istemci uygulamasını oluşturur:

```
CCOMP /in AMQSPUT0/ AMQSPUTC;&
       runnable,systype guardian,nolist,&
       ssv0 "$system.system",&
       ssv1 "MQINSTALLATION_SUBVOL",&
       ld(-LMQINSTALLATION_SUBVOL -lmqic)
```

COBOL programlarının hazırlanması

This topic contains information to consider when you are preparing C programs for the IBM WebSphere MQ client for HP Integrity NonStop Server. OSS ECOBOL derleyicisini kullanırken ve Guardian ECOBOL derleyicisini kullanırken, uygulama oluştururken kullandığınız komutlara ilişkin örnekler içerir.

Kaynak koddan bir COBOL örneği oluşturmak için, ECOBOL derleyicisini kullanın.

The following table lists the libraries that are needed when you are preparing COBOL programs on HP Integrity NonStop Server. Programlarınızı, IBM WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir.

Çizelge 64. . HP Integrity NonStop Server bağlantı kitaplıkları

Kitaplık	**Tanım**
libmqic.so	OSS iş parçacıklı
mqic	Vasi

Bir kuyruk yöneticisine bağlanan bir COBOL uygulaması çalıştırdığınızda, önce *SAVE-ENVIRONMENT* değişkenini 0Ndeğerine ayarlamalısınız. *SAVE-ENVIRONMENT* değişkenini 0N(Açık) olarak ayarlamak için:

- OSS için şu komutu girin:

```
export SAVE-ENVIRONMENT=ON
```

- Guardian için şu komutu girin:

```
param SAVE-ENVIRONMENT ON
```

If you do not set the *SAKLA-ORTAM* variable to AÇIK, when the application attempts to connect to a queue manager, it fails with reason code 2058 (080A) (RC2058): MQRC_Q_MGR_NAME_ERROR.

OSS ECOBOL derleyicisini kullanarak uygulama oluşturulması

Bu bölümde, OSS ECOBOL derleyicisini kullanırken, OSS ya da Guardian için hedeflenen programları oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

Aşağıdaki örnek, bir COBOL istemcisi OSS uygulamasını oluşturur:

```
ecobol -Wsystype=oss
       -Wcobol="ansi;port"
       -Wcobol="consult MQ_INSTALLATION_PATH/opt/mqm/lib/mqicb"
       -Wcopylib=MQ_INSTALLATION_PATH/opt/mqm/inc/cmqcobol.cpy
       -LMQ_INSTALLATION_PATH/opt/mqm/lib -lmqic
       -o amq0put0
       MQ_INSTALLATION_PATH/opt/mqm/samp/amq0put0.cbl
```

Aşağıdaki örnek, bir COBOL istemcisi Guardian uygulamasını oluşturur:

```
ecobol -Wsystype=guardian
       -Wcobol="ansi;port;save all"
       -Wcobol="consult MQ_INSTALLATION_PATH/opt/mqm/lib/mqicb"
       -Wcopylib=MQ_INSTALLATION_PATH/opt/mqm/inc/cmqcobol.cpy
       -LMQ_INSTALLATION_PATH/opt/mqm/lib/G -lmqic
       -o amq0put0
       MQ_INSTALLATION_PATH/opt/mqm/samp/amq0put0.cbl
```

Guardian ECOBOL derleyicisini kullanarak uygulama

Bu bölümde, Guardian ECOBOL derleyicisini kullanırken Guardian için hedeflenen programları oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

MQ_INSTALLATION_SUBVOL , IBM WebSphere MQ ' in kurulu olduğu Guardian birimini ve alt birimi temsil eder.

Aşağıdaki örnek, bir COBOL istemcisi Guardian uygulamasını oluşturur:

```
ECOBOL /in MQSPUTL/ MQSPUT,MQINSTALLATION_SUBVOL.cmqcobol;
       call-shared;ansi;port;save all;nolist;runnable;
       consult MQINSTALLATION_SUBVOL.mqicb;
       eld(-LMQINSTALLATION_SUBVOL -lmqic)
```

pTAL programlarının hazırlanması

HP Integrity NonStop Server platformunda IBM WebSphere MQ istemcisi için pTAL programları oluşturmayı öğrenin.

Kaynak koddan bir pTAL örneği oluşturmak için EPTAL derleyicisini kullanın.

Not:

- pTAL IBM WebSphere MQ uygulamaları, C ya da COBOL dillerinde yazılmış ana bir yordamı kullanmalıdır.
- pTAL uygulamaları yalnızca Guardian 'da oluşturulabilir.

Aşağıdaki çizelgede, HP Integrity NonStop Serverüzerinde pTAL programları hazırlarken gereken kitaplık listelenir. Programlarınızı, IBM WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir.

Çizelge 65. . HP Integrity NonStop Server bağlantı kitaplığı	
Kitaplık	**Tanım**
mqic	Vasi

Guardian EPTAL derleyicisini kullanarak uygulama oluşturulması

Bu bölümde, Guardian EPTAL derleyicisini kullanırken Guardian için hedeflenen programları oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

`MQINSTALLATION_SUBVOL` , IBM WebSphere MQ ' in kurulu olduğu Guardian birimini ve alt birimi temsil eder.

pTAL IBM WebSphere MQ uygulamaları, C ya da COBOL dillerinde yazılmış ana bir yordamı kullanmalıdır.

Aşağıdaki örnek, bir pTAL istemcisi Guardian uygulamasını oluşturur:

```
ASSIGN SSV0, $SYSTEM.SYSTEM
ASSIGN SSV1, MQINSTALLATION_SUBVOL

EPTAL /in MQINSTALLATION_SUBVOL.MQSPUTT/ MQSPUTO;nolist

CCOMP /in MQINSTALLATION_SUBVOL.MQSPTMC/ MQSPUT;
       runnable,systype guardian,extensions,nolist,
       ssv0 "$system.system",
       ssv1 "MQINSTALLATION_SUBVOL",
       eld(MQSPUTO -LMQINSTALLATION_SUBVOL -lmqic)
```

Uygulamanızı HP-UXüzerinde oluşturma

This information describes the additional tasks, and the changes to the standard tasks, that you must perform when building WebSphere MQ for HP-UX applications to run under HP-UX.

C, C + + ve COBOL desteklenmektedir. C++ programlarınızı hazırlamaya ilişkin bilgi için bkz. C++ kullanılması.

HP-UX için WebSphere MQ kullanarak yürütülebilir bir uygulama yaratmak için gerçekleştirmeniz gereken görevler, kaynak kodunuzun yazıldığı programlama diline göre değişir. In addition to coding the MQI calls in your source code, you must add the appropriate language statements to include the WebSphere MQ for HP-UX include files for the language that you are using. Bu dosyaların içeriğini kendinize tanıdık bir hale getiririz. Tam açıklama için “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77 ' e bakın.

Bu konu boyunca, uzun komutları birden çok satırda bölmek için ters eğik çizgi (\) karakteri kullanırız. Bu karakteri girmeyin; her komutu tek bir satır olarak girin.

C programlarının HP-UXiçinde hazırlanması

This topic contains information to consider when preparing C programs in HP-UX; with examples for the IA64 (IPF) platform.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Normal ortamınızda çalışın. Önderlenmiş C programları, *MQ_INSTALLATION_PATH*/samp/bin dizininde sağlanır.

64 bit uygulamalarının programlanması hakkında daha fazla bilgi için 64 bit platformlarda Coding standartlarıbaşlıklı konuya bakın.

SSL kullanmak için, HP-UX üzerinde WebSphere MQ MQI istemcilerinin POSIX iş parçacıkları kullanılarak oluşturulması gerekir.

Dikkate alınacak bazı örnekler şunlardır:

- “IA64 (IPF) platformu” sayfa 422
- “Kitaplıkların bağlanması” sayfa 423

IA64 (IPF) platformu

IA64(IPF) platformunda amqsput0, cliexit ve srvexit örnekleri oluşturun.

The following example builds the sample program amqsput0 as a client application in a non-threaded 32 bit environment:

```
c89 -Wl,+b,: +e -D_HPUX_SOURCE -o amqsputc_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -lmqic
```

The following example builds the sample program amqsput0 as a client application in a threaded 32 bit environment:

```
c89 -mt -Wl,+b,: +e -D_HPUX_SOURCE -o amqsputc_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -lmqic_r -lpthread
```

The following example builds the sample program amqsput0 as a client application in a non-threaded 64 bit environment:

```
c89 +DD64 +e -D_HPUX_SOURCE -o amqsputc_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqic
```

Aşağıdaki örnek, örnek programı amqsput0 örnek programını, iş parçacıklı bir 64 bit ortamında istemci uygulaması olarak oluşturur:

```
c89 -mt +DD64 +e -D_HPUX_SOURCE -o amqsputc_64_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqic_r -lpthread
```

The following example builds the sample program amqsput0 as a server application in a non-threaded 32 bit environment:

```
c89 -Wl,+b,: +e -D_HPUX_SOURCE -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -lmqm
```

Aşağıdaki örnek, örnek programı amqsput0 örnek programını, iş parçacıklı 32 bit ortamında sunucu uygulaması olarak oluşturur:

```
c89 -mt -Wl,+b,: +e -D_HPUX_SOURCE -o amqsput_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -lmqm_r -lpthread
```

Aşağıdaki örnek, örnek programı amqsput0 örnek programını iş parçacıklı bir 64 bit ortamında sunucu uygulaması olarak oluşturur:

```
c89 +DD64 +e -D_HPUX_SOURCE -o amqsput_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqm
```

Aşağıdaki örnek, örnek programı amqsput0 örnek programını, iş parçacıklı bir 64 bit ortamında sunucu uygulaması olarak oluşturur:

```
c89 -mt +DD64 +e -D_HPUX_SOURCE -o amqsput_64_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
  -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqm_r -lpthread
```

Aşağıdaki örnek, iş parçacıklı olmayan 32 bit ortamında bir istemci çıkış klipini oluşturur:

```
c89 +e +z -c -D_HPUX_SOURCE -o cliexit.o cliexit.c -IMQ_INSTALLATION_PATH/inc
ld +b: -b cliexit.o +ee MQStart -o /var/mqm/exits/cliexit_32 -LMQ_INSTALLATION_PATH/lib \
-L/usr/lib/hpux32 -lmqic
```

Aşağıdaki örnek, iş parçacıklı 32 bit ortamında bir istemci çıkışı iklizimi oluşturur:

```
c89 -mt +e +z -c -D_HPUX_SOURCE -o cliexit.o cliexit.c -IMQ_INSTALLATION_PATH/inc
ld +b: -b cliexit.o +ee MQStart -o /var/mqm/exits/cliexit_32_r -LMQ_INSTALLATION_PATH/lib \
-L/usr/lib/hpux32 -lmqic_r -lpthread
```

Aşağıdaki örnek, iş parçacıklı olmayan 64 bit ortamında bir istemci çıkış klipini oluşturur:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o cliexit.o cliexit.c -IMQ_INSTALLATION_PATH/inc
ld -b cliexit.o +ee MQStart -o /var/mqm/exits64/cliexit_64 \
-LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqic
```

Aşağıdaki örnek, yivli 64 bit ortamında bir istemci çıkış klipini oluşturur:

```
c89 -mt +DD64 +e +z -c -D_HPUX_SOURCE -o cliexit.o cliexit.c -IMQ_INSTALLATION_PATH/inc
ld  -b cliexit.o +ee MQStart -o /var/mqm/exits/cliexit_64_r \
-LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqic_r -lpthread
```

Aşağıdaki örnek, iş parçacıklı olmayan 32 bit ortamında bir sunucu çıkışı srvexit oluşturmasını sağlar:

```
c89 +e +z -c -D_HPUX_SOURCE -o srvexit.o srvexit.c -IMQ_INSTALLATION_PATH/inc
ld +b: -b srvexit.o +ee MQStart -o /var/mqm/exits/srvexit_32 -LMQ_INSTALLATION_PATH/lib \
-L/usr/lib/hpux32 -lmqm
```

Aşağıdaki örnek, iş parçacıklı 32 bit ortamında bir sunucu çıkışı srvexit oluşturmasını sağlar:

```
c89 -mt +e +z -c -D_HPUX_SOURCE -o srvexit.o srvexit.c -IMQ_INSTALLATION_PATH/inc
ld +b: -b srvexit.o +ee MQStart -o /var/mqm/exits/srvexit_32_r -LMQ_INSTALLATION_PATH/lib \
-L/usr/lib/hpux32 -lmqm_r -lpthread
```

Aşağıdaki örnek, iş parçacıklı olmayan 64 bit ortamında bir sunucu çıkışı srvexit oluşturmasını sağlar:

```
c89 +DD64 +e +z -c -D_HPUX_SOURCE -o srvexit.o srvexit.c
-IMQ_INSTALLATION_PATHMQ_INSTALLATION_PATH/inc
ld -b srvexit.o +ee MQStart -o /var/mqm/exits64/srvexit_64 \
-LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqm
```

Aşağıdaki örnek, yivli bir 64 bit ortamında bir sunucu çıkışı srvexit 'i oluşturur:

```
c89 -mt +DD64 +e +z -c -D_HPUX_SOURCE -o srvexit.o srvexit.c -IMQ_INSTALLATION_PATH/inc
ld  -b srvexit.o +ee MQStart -o /var/mqm/exits/srvexit_64_r \
-LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -lmqm_r -lpthread
```

Kitaplıkların bağlanması

Programlarınızı, WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir.

Aşağıdaki çizelge, farklı ortamlarda kullanılacak kitaplığı göstermektedir.

Donanım platformu	İş parçacıklı ya da iş parçacıklı ortam	Program/çıkış tipi	Kitaplık dosyası
IA64 (IPF)	Işıklı	C için Sunucu ve İstemci	libmqm_r.so
IA64 (IPF)	Işıklı	C İçin İstemci	libmqic_r.so
IA64 (IPF)	İş parçacıklı olmayan	C için Sunucu ve İstemci	libmqm.so
IA64 (IPF)	İş parçacıklı olmayan	C İçin İstemci	libmqic.so

Not:

1. Kurulabilir bir hizmet yazıyorsanız (ek bilgi için Yönetme başlıklı konuya bakın), `libmqmzf.sl` kitaplığına bağlanmanız gerekir.
2. If you are producing an application for external coordination by an XA-compliant transaction manager such as IBM TXSeries Encina, or BEA Tuxedo, you need to link to the `libmqmxa.sl` (or `libmqmxa64.sl` if your transaction manager treats the 'long' type as 64 bit) and `libmqz.sl` libraries in a non-threaded application and to the `libmqmxa_r.sl` (or `libmqmxa64_r.sl`) and `libmqz_r.sl` libraries in a threaded application.
3. Diğer ürün kitaplıklarından önce WebSphere MQ kitaplıklarını bağlamanız gerekir.

COBOL programlarının HP-UXiçinde hazırlanması

Learn about preparing COBOL programs in HP-UX, using Micro Focus Server Express with WebSphere MQ on the IA64 (IPF) platform, and running programs in the WebSphere MQ MQI client environment.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Notes kullanıcıları

1. 32 bit COBOL kopya kitapları aşağıdaki dizine takılır:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy32
   ```

   ve simgesel bağlantılar aşağıdaki yerde yaratılır:

   ```
   MQ_INSTALLATION_PATH/inc
   ```

2. 64 bit COBOL kopya kitapları aşağıdaki dizine takılır:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy64
   ```

3. Aşağıdaki örneklerde COBCPY to:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy32
   ```

   32 bit uygulamalar için ve:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy64
   ```

   64 bit uygulamalar için.

Micro Focus derleyicisini kullanarak programları derleyin. Yapıları bildiren kopya dosyaları *MQ_INSTALLATION_PATH*/inc. içinde yer alan kopyalardır:

```
$ export LIB=MQ_INSTALLATION_PATH/lib:$LIB
$ export COBCPY="<COBCPY>"
```

32 bit program derleniyor:

```
$ cob32 -xv amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb Server for COBOL
$ cob32 -xv amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb Client for COBOL
$ cob32 -xtv amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb_r Threaded Server for COBOL
$ cob32 -xtv amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb_r Threaded Client for COBOL
```

64 bit programları derleniyor:

```
$ cob64 -xv amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb Server for COBOL
$ cob64 -xv amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb Client for COBOL
```

```
$ cob64 -xtv amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb_r Threaded Server for COBOL
$ cob64 -xtv amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb_r Threaded Client for COBOL
```

Burada `amqsput` , örnek bir programdır

Yürütme ortamı yığını büyüklüklerinin yeterli olduğunu doğrulayın; önerilen alt sınır 16 KB 'dir.

Programlarınızı, WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir. Aşağıdaki çizelge, farklı ortamlarda kullanılacak kitaplığı göstermektedir.

Donanım platformu	Program/çıkış tipi	Kitaplık dosyası
IA64 (IPF)	COBOL için sunucu	libmqmcb.so
IA64 (IPF)	COBOL için İstemci	libmqicb.so
IA64 (IPF)	İş parçacığı uygulamaları	libmqmcb_r.so

Using Micro Focus Server Express with WebSphere MQ on the IA64 (IPF) platform

HP/IPF platformundaki WebSphere MQ ile birlikte Micro Focus Server Express ürününün kullanılmasıyla ilgili ayrıntılı bilgi için “Address Space models supported by WebSphere MQ for HP-UX on IA64 (IPF)” sayfa 426 konusuna bakın.

Programs to run in the WebSphere MQ MQI client environment

MQI istemcinizi bir sunucuya bağlamak için LU 6.2 'u kullanıyorsanız, uygulamanızı SNAplusAPI ürününün bir parçası olan libsna.a' a bağlayın. Derleme ve bağlantı komutunuzda `-lV3` ve `-lstr` seçeneklerini kullanın.

- `-lV3` seçeneği, AT & T sinyalizasyon kitaplığına program erişiminizi sağlar (SNAplusAPI AT & T sinyallerini kullanır)
- `-lstr` seçeneği, programınızı akışlar bileşenine bağlar

Preparing CICS programs in HP-UX

HP-UX' de CICS hareket programları oluşturmayı öğrenin.

Örnek CICS hareketini oluşturmak için amqscic0.ccsaşağıdaki komutu çalıştırın:

```
$ export USERLIB="-lmqm_r"
$ cicstcl -l C amqscic0.ccs
```

CICS ' i WebSphere MQile bağlamanızı sağlamak için XA anahtar birimi sağlanmıştır:

Çizelge 66. CICS uygulamaları için temel kod (HP-UX)		
Tanım	**C (kaynak)**	**C (exec)**
XA kullanıma hazırlama yordamı	amqzscix.c	amqzsc

You can find more information about supporting CICS transactions in the Yönetme.

TXSeries CICS desteği

HP-UX üzerinde WebSphere MQ , XA arabirimini kullanarak TXSeries CICS ' u destekler. CICS uygulamalarının, MQ kitaplıklarının iş parçacıklı sürümüyle bağlantılı olduğundan emin olun.

C ya da COBOL ' de aynı CICS bölgesine yüklenen WebSphere MQ programları yazın. Aynı CICS bölgesine C ve COBOL MQI çağrılarının birleşiminden oluşan bir birleşim yapamazsınız. Most MQI calls in the second language used fail with a reason code of `MQRC_HOBJ_ERROR`.

CICS C örnek hareketi

Bir CICS WebSphere MQ işlemi için Örnek C kaynağı, AMQSCIC0.CCS. Hareket, varsayılan kuyruk yöneticisinde SYSTEM.SAMPLE.CICS.WORKQUEUE iletim kuyruğundan iletileri okur ve bunları, iletinin iletim üstbilgisinde yer alan kuyruk adıyla yerel kuyruğa yerleştirir. SYSTEM.SAMPLE.CICS.DLQ. Örnek MQSC komut kütüğünü (AMQSCIC0.TST , bu kuyrukları ve örnek giriş kuyruklarını yaratmanızı sağlar.

Preparing CICS COBOL programs using Micro Focus COBOL

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Micro Focus COBOL olanağını kullanmak için aşağıdaki adımları izleyin:

1. Aşağıdaki komutu kullanarak, WebSphere MQ COBOL Runtime kitaplık modülünü yürütme ortamı kitaplığına ekleyin:

   ```
   cicsmkcobol -L/usr/lib/dce -LMQ_INSTALLATION_PATH/lib  \
               MQ_INSTALLATION_PATH/lib/libmqmcbrt.o -lmqe_r
   ```

   Not: `cicsmkcobol`ile, WebSphere MQ , C programlama dilinde COBOL uygulamanızın MQI çağrılarını gerçekleştirmenize izin vermez.

   Var olan uygulamalarınızda bu tür çağrılar varsa, bu işlevleri COBOL uygulamalarından kendi kitaplığınıza taşımanız önerilir; örneğin, `myMQ.so`. After moving these functions do not include the WebSphere MQ library `libmqmcbrt.o` when building the COBOL application for CICS.

   Ayrıca, COBOL uygulamanızın COBOL MQI çağrısı yapmazsa, `libmqmz_r` ile `cicsmkcobol`bağlantısını bağlamayın.

   Bu, Micro Focus COBOL dil yöntemi dosyasını oluşturur ve CICS Runtime COBOL kitaplığının UNIX and Linux systemssistemlerinde WebSphere MQ ' u aramasını sağlar.

   Not: `cicsmkcobol` komutunu yalnızca aşağıdaki ürünlerden birini kurduğunuzda çalıştırın:

   - Micro Focus COBOL ' in yeni sürümü veya
   - New version or release of CICS for HP-UX
   - Desteklenen herhangi bir veritabanı ürününün yeni sürümü ya da yayın düzeyi (yalnızca COBOL işlemleri için)
   - New version or release of WebSphere MQ
2. Aşağıdaki ortam değişkenini dışa aktarın:

   ```
   COBCPY=MQ_INSTALLATION_PATH/inc  export COBCPY
   ```
3. Programı çevirerek, derleyin ve bağlayın:

   ```
   cicstcl -l COBOL -e <yourprog>.ccp
   ```

Address Space models supported by WebSphere MQ for HP-UX on IA64 (IPF)

HP-UX , WebSphere MQ uygulamaları tarafından kullanılabilecek çeşitli adres alanı modelleri sağlar.

HP-UX , iki Adres Alanı modelini destekler:

- MGAS-çoğunlukla Genel Adres alanı (varsayılan değer budur ve WebSphere MQtarafından kullanılır)
- MPAS-çoğunlukla Özel Adres alanı

WebSphere MQ ' ya bağlanan uygulamalar, MGAS ya da MPAS adres alanı modellerini kullanabilir. Applications built using the MPAS model that connect to WebSphere MQ using shared memory might incur a minor performance cost due to the inefficiency in mapping the shared memory pages used by WebSphere MQ into the virtual address space of the MPAS program.

Micro Focus Server Express kullanılarak oluşturulan COBOL uygulamaları varsayılan olarak MPAS modelini kullanır.

Bir program tarafından kullanılan adresleme modelini denetlemek ve değiştirmek için **chatr** programını kullanabilirsiniz.

32 bit MPAS programlarından WebSphere MQ ' ya bağlanılırken sorunlarla karşılaşırsanız, MGAS adresleme modelini kullanmayı düşünün ya da uygulamanızı 32 bitlik MPAS uygulaması yerine 64 bit MPAS uygulaması olarak kullanın.

MGAS ve MPAS adres alanı modellerine ilişkin ek ayrıntılar HP-UX belgelerinde bulunabilir.

Uygulamanızı Linuxüzerinde oluşturma

This information describes the additional tasks, and the changes to the standard tasks, that you must perform when building WebSphere MQ for Linux applications to run.

C ve C++ desteklenir. C++ programlarınızı hazırlamaya ilişkin bilgi için bkz. C++ kullanılması.

C programlarını Linuxiçinde hazırlama

Önderlenmiş C programları, *MQ_INSTALLATION_PATH*/samp/bin dizininde sağlanır. Kaynak koddan bir örnek oluşturmak için, gcc derleyicisini kullanın.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Normal ortamınızda çalışın. 64 bit kullanan uygulamalar hakkında daha fazla bilgi için bkz. 64 bit altyapılarda koşma standartları.

Kitaplıkların bağlanması

The following tables lists the libraries that are needed when preparing C programs on Linux.

- Programlarınızı, WebSphere MQtarafından sağlanan uygun kitaplıkla ilişkilendirmeniz gerekir.

  İş parçacıklı olmayan bir ortamda, aşağıdaki kitaplıklardan birine bağlayın:

Kitaplık dosyası	Program/çıkış tipi
libmqm.so	C sunucusu için sunucu
libmqic.so & libmqm.so	C İçin İstemci

  İş parçacıklı bir ortamda, aşağıdaki kitaplıklardan birine bağlantı:

Kitaplık dosyası	Program/çıkış tipi
libmqm_r.so	C sunucusu için sunucu
libmqic_r.so & libmqm_r.so	C İçin İstemci

Not:

1. Kurulabilir bir hizmet yazıyorsanız (ek bilgi için Yönetme başlıklı konuya bakın), libmqmzf.so kitaplığına bağlanmanız gerekir.
2. If you are producing an application for external coordination by an XA-compliant transaction manager such as IBM TXSeries Encina, or BEA Tuxedo, you need to link to the libmqmxa.so (or libmqmxa64.so if your transaction manager treats the 'long' type as 64 bit) and libmqz.so libraries in a non-threaded application and to the libmqmxa_r.so (or libmqmxa64_r.so) and libmqz_r.so libraries in a threaded application.
3. Diğer ürün kitaplıklarından önce WebSphere MQ kitaplıklarını bağlamanız gerekir.

31 bit uygulamalar oluşturuluyor

Bu konu, çeşitli ortamlarda 31 bit programları oluşturmak için kullanılan komutlara ilişkin örnekleri içerir.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C istemci uygulaması, 31-bit, yivsiz

```
gcc -m31  -o famqsputc_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqic
```

C istemcisi uygulaması, 31-bit, yivli

```
gcc -m31  -o amqsputc_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqic_r -lpthread
```

C sunucusu uygulaması, 31-bit, iş parçacıklı olmayan

```
gcc -m31  -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqm
```

C sunucusu uygulaması, 31-bit, iş parçacığı

```
gcc -m31  -o amqsput_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqm_r -lpthread
```

C++ istemci uygulaması, 31-bit, iş parçacıklı olmayan

```
g++ -m31 -fsigned-char  -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl
-limqb23gl -lmqic
```

C++ istemci uygulaması, 31-bit, iş parçacığı

```
g++ -m31 -fsigned-char  -o imqsputc_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl_r
-limqb23gl_r -lmqic_r -lpthread
```

C++ sunucu uygulaması, 31-bit, iş parçacıklı olmayan

```
g++ -m31 -fsigned-char  -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl
-limqb23gl -lmqm
```

C++ sunucu uygulaması, 31-bit, iş parçacığı

```
g++ -m31 -fsigned-char  -o imqsput_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl_r
-limqb23gl_r -lmqm_r -lpthread
```

C istemci çıkışı, 31-bit, yivsiz

```
gcc -m31 -shared -fPIC  -o /var/mqm/exits/cliexit_32 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
-lmqic
```

C istemcisi çıkışı, 31-bit, yivli

```
gcc -m31 -shared -fPIC  -o /var/mqm/exits/cliexit_32_r cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
```

```
-Wl,-rpath=/usr/lib
-lmqic_r -lpthread
```

C sunucusu çıkışı, 31-bit, iş parçacıklı olmayan

```
gcc -m31 -shared -fPIC  -o /var/mqm/exits/srvexit_32 srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
-lmqm
```

C sunucusu çıkışı, 31-bit, yivli

```
gcc -m31 -shared -fPIC  -o /var/mqm/exits/srvexit_32_r srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
-lmqm_r -lpthread
```

32 bitlik uygulamalar oluşturuluyor

Bu konu, çeşitli ortamlarda 32 bit programları oluşturmak için kullanılan komutlara ilişkin örnekleri içerir.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C istemcisi uygulaması, 32 bit, iş parçacıklı olmayan

```
gcc -m32 -o amqsputc_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqic
```

C istemci uygulaması, 32 bit, yivli

```
gcc -m32 -o amqsputc_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqic_r -lpthread
```

C sunucusu uygulaması, 32 bitlik, iş parçacıklı olmayan

```
gcc -m32 -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqm
```

C sunucusu uygulaması, 32 bit, yivli

```
gcc -m32 -o amqsput_32_r amqsput0.c -IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqm_r -lpthread
```

C++ istemci uygulaması, 32 bit, iş parçacıklı olmayan

```
g++ -m32 -fsigned-char -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib  -Wl,-rpath=/usr/lib
-limqc23gl -limqb23gl -lmqic
```

C++ istemci uygulaması, 32 bit, yivli iş parçacığı

```
g++ -m32 -fsigned-char -o imqsputc_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib  -Wl,-rpath=/usr/lib
-limqc23gl_r -limqb23gl_r -lmqic_r -lpthread
```

C++ sunucu uygulaması, 32 bitlik, iş parçacıklı olmayan

```
g++ -m32 -fsigned-char -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl -limqb23gl -lmqm
```

C++ sunucu uygulaması, 32 bit, yivli iş parçacığı

```
g++ -m32 -fsigned-char -o imqsput_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl_r -limqb23gl_r -lmqm_r -lpthread
```

C istemci çıkışı, 32 bit, yivsiz olmayan

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/cliexit_32 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
-lmqic
```

C istemcisi çıkışı, 32 bit, yivli iş parçacığı

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/cliexit_32_r cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
-lmqic_r -lpthread
```

C sunucusu çıkışı, 32 bitlik, iş parçacıklı olmayan

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/srvexit_32 srvexit.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -lmqm
```

C sunucusu çıkışı, 32 bit, yivli iş parçacığı

```
gcc -m32 -shared -fPIC -o /var/mqm/exits/srvexit_32_r srvexit.c
IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
-Wl,-rpath=/usr/lib
lmqm_r -lpthread
```

64 bit uygulamalar oluşturuluyor

Bu konu, çeşitli ortamlarda 64 bit programları oluşturmak için kullanılan komutlara ilişkin örnekler içerir.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C istemcisi uygulaması, 64 bitlik, iş parçacıklı olmayan

```
gcc -m64  -o amqsputc_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -lmqic
```

C istemcisi uygulaması, 64 bitlik, iş parçacıklı

```
gcc -m64  -o amqsputc_64_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -lmqic_r
-lpthread
```

C sunucusu uygulaması, 64 bitlik, iş parçacıklı olmayan

```
gcc -m64  -o amqsput_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -lmqm
```

C sunucusu uygulaması, 64 bitlik, iş parçacıklı

```
gcc -m64  -o amqsput_64_r amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -lmqm_r
-lpthread
```

C++ istemci uygulaması, 64 bitlik, iş parçacıklı olmayan

```
g++ -m64 -fsigned-char -o imqsputc_64 imqsput.cpp
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqc23gl -limqb23gl -lmqic
```

C++ istemci uygulaması, 64 bitlik, iş parçacıklı

```
g++ -m64 -fsigned-char -o imqsputc_64_r imqsput.cpp
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqc23gl_r -limqb23gl_r -lmqic_r -lpthread
```

C++ sunucu uygulaması, 64 bitlik, iş parçacıklı olmayan

```
g++ -m64 -fsigned-char  -o imqsput_64 imqsput.cpp
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -limqs23gl -limqb23gl -lmqm
```

C++ sunucu uygulaması, 64 bitlik, iş parçacıklı

```
g++ -m64 -fsigned-char  -o imqsput_64_r imqsput.cpp
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -limqs23gl_r -limqb23gl_r -lmqm_r -lpthread
```

C istemci çıkışı, 64 bitlik, iş parçacıklı olmayan

```
gcc -m64 -shared -fPIC  -o /var/mqm/exits64/cliexit_64 cliexit.c
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -lmqic
```

C istemcisi çıkışı, 64 bit, yivli

```
gcc -m64 -shared -fPIC  -o /var/mqm/exits64/cliexit_64_r cliexit.c
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -lmqic_r -lpthread
```

C sunucusu çıkışı, 64 bitlik, iş parçacıklı olmayan

```
gcc -m64 -shared -fPIC  -o /var/mqm/exits64/srvexit_64 srvexit.c
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -lmqm
```

C sunucusu çıkışı, 64 bitlik, iş parçacıklı

```
gcc -m64 -shared -fPIC  -o /var/mqm/exits64/srvexit_64_r srvexit.c
-IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64
-Wl,-rpath=/usr/lib64 -lmqm_r -lpthread
```

Preparing COBOL programs in Linux

COBOL programlarının Linux içinde hazırlanması ve Micro Focus COBOL kullanarak COBOL programlarının hazırlanması hakkında bilgi edinin.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

1. 32 bit COBOL kopya kitapları aşağıdaki dizine takılır:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy32
   ```

   ve simgesel bağlantılar aşağıdaki yerde yaratılır:

   ```
   MQ_INSTALLATION_PATH/inc
   ```

2. 64 bit altyapılarda, 64 bit COBOL kopya kitapları aşağıdaki dizine kurulur:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy64
   ```

3. Aşağıdaki örneklerde COBCPY to:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy32
   ```

   32 bit uygulamalar için ve:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy64
   ```

   64 bit uygulamalar için.

Programınızı aşağıdaki bağlantılardan biriyle ilişkilendirmeniz gerekir:

Kitaplık dosyası	Program/çıkış tipi
libmqmcb.so	COBOL için sunucu
libmqicb.so	COBOL için İstemci
libmqmcb_r.so	COBOL için sunucu (iş parçacıklı uygulama)
libmqicb_r.so	COBOL için İstemci (iş parçacıklı uygulama)

COBOL programlarının Micro Focus COBOL

Programınızı derlemeden önce ortam değişkenlerini aşağıdaki gibi ayarlayın:

```
export COBCPY=<COBCPY>
export LIB=MQ_INSTALLATION_PATHlib:$LIB
```

Desteklenen, Micro Focus COBOL kullanarak desteklenen 32 bit COBOL programını derlemek için şunu girin:

```
  $ cob32 -xvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb Server for COBOL
  $ cob32 -xvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb Client for COBOL
  $ cob32 -xtvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb_r Threaded Server for COBOL
  $ cob32 -xtvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb_r Threaded Client for COBOL
```

Bir 64 bit COBOL programını Micro Focus COBOL kullanarak derlemek için şunu girin:

```
  $ cob64 -xvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb Server for COBOL
```

```
$ cob64 -xvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb Client for COBOL
$ cob64 -xtvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb_r Threaded Server for COBOL
$ cob64 -xtvP amqsput.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb_r Threaded Client for COBOL
```

Burada `amqsput` , örnek bir programdır

Gereksinim duyardığınız ortam değişkenlerine ilişkin açıklamalar için Micro Focus COBOL belgelerine bakın.

Uygulamanızı Solaris üzerinde oluşturma

Bu bilgiler, Solaris uygulamaları altında çalışmak üzere WebSphere MQ oluştururken gerçekleştirmeniz gereken ek görevleri ve standart görevlerle ilgili değişiklikleri açıklar.

COBOL, C ve C++ programlama dilleri desteklenmektedir. C++ programlarınızı hazırlamaya ilişkin bilgi için bkz. C++ kullanılması.

Kaynak kodunuzda MQI çağrılarını kodlamaya ek olarak, uygun içerme dosyalarını eklemelisiniz. Bu dosyaların içeriğini kendinize tanıdık bir hale getiriniz. Tam açıklama için “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77 ' e bakın.

Bu konu boyunca ters eğik çizgi (\) karakteri, uzun komutları birden çok satıra bölmek için kullanılır. Bu karakteri girmeyin, her komutu tek bir satır olarak girin.

C programlarını Solaris 'e hazırlama

Önderlenmiş C programları, *MQ_INSTALLATION_PATH*/`samp/bin` dizininde sağlanır.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

64 bit kullanan uygulamalar hakkında daha fazla bilgi için bkz. 64 bit altyapılarda koşma standartları.

Solaris 'e ilişkin yalnızca WebSphere MQ MQI istemcisi kurulu bir makineyle ilgili programları kullanmak istiyorsanız, bu programları, bunları istemci kitaplığıyla (`-lmqic`) ilişkilendirecek şekilde derleyin.

If you use the unsupported compiler `?usr?ucb?cc`, your application might compile and link successfully. Ancak, uygulamayı çalıştırdığınızda, kuyruk yöneticisine bağlanma girişiminde bulunduğunda başarısız olur.

Not: 32 bit Solaris x86 SSL ve FIPS 140-2 uyumlu işletim için yapılandırılan TLS istemcileri, Intel sistemlerinde çalışırken başarısız olur. Bu hata, FIPS 140-2 uyumlu GSKit-Crypto Solaris x86 32 bit kitaplık dosyasının Intel yonga setine yüklenmediği için ortaya çıkar. Etkilenen sistemlerde, istemci hata günlüğünde AMQ9655 hatası raporlanır. Bu sorunu çözmek için FIPS 140-2 uyumluluğunu geçersiz kılın ya da 64 bit kodu etkilenmediği için istemci uygulamasının 64 bit 'i yeniden derleyin.

Kitaplıkların bağlanması

Uygulama tipiniz için uygun olan WebSphere MQ kitaplıklarıyla bağlantı vermelisiniz:

Kitaplık dosyaları	Program/çıkış tipi
libmqm.so	C sunucusu için sunucu
libmqic.so & libmqm.so	C İçin İstemci

Not:

1. Kurulabilir bir hizmet yazıyorsanız (daha fazla bilgi için bkz. Yönetme), `libmqmzf.so` kitaplığına bağlantı.
2. If you are producing an application for external coordination by an XA-compliant transaction manager such as IBM TXSeries Encina, or BEA Tuxedo, you must link to the `libmqmxa.so` (or `libmqmxa64.so` if your transaction manager treats the 'long' type as 64 bit) and `libmqz.so` libraries.
3. Diğer ürün kitaplıklarından önce WebSphere MQ kitaplıklarını bağlamanız gerekir.

Building applications on x86-64

Bu konuda, x86-64 platformunda çeşitli ortamlarda programlar oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C istemci uygulaması, 32 bit

```
cc -xarch=386 -mt -o amqsputc_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -lmqic -lsocket -lnsl -ldl
```

C istemci uygulaması, 64 bit

```
cc -xarch=amd64 -mt -o amqsputc_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -lmqic -lsocket
-lnsl -ldl
```

C sunucusu uygulaması, 32 bit

```
cc -xarch=386 -mt -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -lmqm -lsocket -lnsl -ldl
```

C sunucusu uygulaması, 64 bit

```
cc -xarch=amd64 -mt -o amqsput_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -lmqm -lsocket
-lnsl -ldl
```

C++ istemci uygulaması, 32 bit

```
CC -xarch=386 -mt -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqc23as -limqb23as -lmqic -lsocket -lnsl -ldl
```

C++ istemci uygulaması, 64 bit

```
CC -xarch=amd64 -mt -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqc23as
-limqb23as
-lmqic -lsocket -lnsl -ldl
```

C++ sunucu uygulaması, 32 bit

```
CC -xarch=386 -mt -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqs23as -limqb23as -lmqm
-lsocket -lnsl -ldl
```

C++ sunucu uygulaması, 64 bit

```
CC -xarch=amd64 -mt -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqs23as
-limqb23as -lmqm
-lsocket -lnsl -ldl
```

C istemcisi çıkışı, 32 bit

```
cc -xarch=386 -mt -G -KPIC -o /var/mqm/exits/cliexit_32 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib
```

```
-R/usr/lib/32 -lmqic
-lsocket -lnsl -ldl
```

C istemci çıkışı, 64 bit

```
cc -xarch=amd64 -mt -G -KPIC -o /var/mqm/exits64/cliexit_64 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64
-R/usr/lib/64 -lmqic
-lsocket -lnsl -ldl
```

C sunucusu çıkışı, 32 bit

```
cc -xarch=386 -mt -G -KPIC -o /var/mqm/exits/srvexit_32 srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib
-R/usr/lib/32 -lmqm
-lsocket -lnsl -ldl
```

C sunucusu çıkışı, 64 bit

```
cc -xarch=amd64 -mt -G -KPIC -o /var/mqm/exits64/srvexit_64 srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64
-R/usr/lib/64 -lmqm
-lsocket -lnsl -ldl
```

SPARC üzerinde uygulama oluşturma

Bu konuda, SPARC platformunda çeşitli ortamlarda programlar oluşturmak için kullanılan komutlara ilişkin örnekler yer alır.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C istemci uygulaması, 32 bit

```
cc -xarch=v8plus -mt -o amqsputc_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -lmqic -lsocket -lnsl -ldl
```

C istemci uygulaması, 64 bit

```
cc -xarch=v9 -mt -o amqsputc_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -lmqic
-lsocket -lnsl -ldl
```

C sunucusu uygulaması, 32 bit

```
cc -xarch=v8plus -mt -o amqsput_32 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -lmqm -lsocket -lnsl -ldl
```

C sunucusu uygulaması, 64 bit

```
cc -xarch=v9 -mt -o amqsput_64 amqsput0.c -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -lmqm
-lsocket -lnsl -ldl
```

C++ istemci uygulaması, 32 bit

```
CC -xarch=v8plus -mt -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqc23as -limqb23as
-lmqic
-lsocket -lnsl -ldl
```

C++ istemci uygulaması, 64 bit

```
CC -xarch=v9 -mt -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqc23as
-limqb23as
-lmqic -lsocket -lnsl -ldl
```

C++ sunucu uygulaması, 32 bit

```
CC -xarch=v8plus -mt -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib
-RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqs23as -limqb23as -lmqm
-lsocket -lnsl -ldl
```

C++ sunucu uygulaması, 64 bit

```
CC -xarch=v9 -mt -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqs23as
-limqb23as -lmqm
-lsocket -lnsl -ldl
```

C istemcisi çıkışı, 32 bit

```
cc -xarch=v8plus -mt -G -KPIC -o /var/mqm/exits/cliexit_32 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib
-R/usr/lib/32 -lmqic
-lsocket -lnsl -ldl
```

C istemci çıkışı, 64 bit

```
cc -xarch=v9 -mt -G -KPIC -o /var/mqm/exits64/cliexit_64 cliexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64
-R/usr/lib/64 -lmqic
-lsocket -lnsl -ldl
```

C sunucusu çıkışı, 32 bit

```
cc -xarch=v8plus -mt -G -KPIC -o /var/mqm/exits/srvexit_32 srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib
-R/usr/lib/32 -lmqm
-lsocket -lnsl -ldl
```

C sunucusu çıkışı, 64 bit

```
cc -xarch=v9 -mt -G -KPIC -o /var/mqm/exits64/srvexit_64 srvexit.c
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64
-R/usr/lib/64 -lmqm
-lsocket -lnsl -ldl
```

COBOL programlarının Solaris 'e hazırlanması

COBOL programlarının Solaris 'e hazırlanmasına ilişkin bilgi edinin.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

1. 32 bit COBOL kopya kitapları aşağıdaki dizine takılır:

   ```
   MQ_INSTALLATION_PATH/inc/cobcpy32
   ```

   ve simgesel bağlantılar aşağıdaki yerde yaratılır:

   ```
   MQ_INSTALLATION_PATH/inc
   ```

2. 64 bit COBOL kopya kitapları aşağıdaki dizine takılır:

```
MQ_INSTALLATION_PATH/inc/cobcpy64
```

3. Aşağıdaki örneklerde COBCPY to:

```
MQ_INSTALLATION_PATH/inc/cobcpy32
```

   32 bit uygulamalar için ve:

```
MQ_INSTALLATION_PATH/inc/cobcpy64
```

   64 bit uygulamalar için.

Micro Focus derleyicisini kullanarak programları derleyin. Yapıları bildiren kopya dosyaları *MQ_INSTALLATION_PATH*/inciçinde yer alıyor:

```
$ export LIB=MQ_INSTALLATION_PATH/lib:$LIB
$ export COBCPY="<COBCPY>"
```

32 bit program derleniyor:

- `$ cob32 -xv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb`

  COBOL için sunucu
- `$ cob32 -xv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb`

  COBOL için İstemci
- `$ cob32 -xtv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqmcb_r`

  COBOL İçin İş Y
- `$ cob32 -xtv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib -lmqicb_r`

  COBOL için İş parçacığı

64 bit programlar derleniyor:

- `$ cob64 -xv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb`

  COBOL için sunucu
- `$ cob64 -xv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb`

  COBOL için İstemci
- `$ cob64 -xtv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqmcb_r`

  COBOL İçin İş Y
- `$ cob64 -xtv amqs0put0.cbl -L MQ_INSTALLATION_PATH/lib64 -lmqicb_r`

  COBOL için İş parçacığı

Burada *amqs0put0.cbl* örnek bir programdır.

Programınızı aşağıdakilerden biriyle bağlamanız gerekir:

- libmqmcb.so

  COBOL için sunucu
- libmqicb.so

  COBOL için İstemci

Solaris 'te CICS programlarının hazırlanması

Solaris 'te CICS programlarının hazırlanmasına ilişkin bilgi edinin.

CICS ' i WebSphere MQile bağlamanızı sağlamak için XA anahtar birimi sağlanmıştır:

Çizelge 67. CICS uygulamaları için temel kod (Solaris)		
Tanım	**C (kaynak)**	**C (exec)**
XA kullanıma hazırlama yordamı	amqzscix.c	amqzsc- Solaris için TXSeries

İşlemlerinizi her zaman iş parçacığı güvenli WebSphere MQ kitaplığı libmqm.soile bağlantılayın.

You can find more information about supporting CICS transactions in the Yönetme.

TXSeries CICS desteği

Solaris için WebSphere MQ , XA arabirimini kullanarak TXSeries CICS ' u destekler.

C ya da COBOL ' de aynı CICS bölgesine yüklenen WebSphere MQ programları yazın. Aynı CICS bölgesine C ve COBOL MQI çağrılarının birleşiminden oluşan bir birleşim yapamazsınız. Most MQI calls in the second language used fail with a reason code of `MQRC_HOBJ_ERROR`.

Preparing CICS COBOL programs using Micro Focus COBOL

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Micro Focus COBOL olanağını kullanmak için aşağıdaki adımları izleyin:

1. Aşağıdaki komutu kullanarak, WebSphere MQ COBOL Runtime kitaplık modülünü yürütme ortamı kitaplığına ekleyin:

   ```
   cicsmkcobol -L/usr/lib/dce -LMQ_INSTALLATION_PATH/lib  \
               MQ_INSTALLATION_PATH/lib/libmqmcbrt.o -lmqe
   ```

   Not: `cicsmkcobol`ile, WebSphere MQ , C programlama dilinde COBOL uygulamanızın MQI çağrılarını gerçekleştirmenize izin vermez.

   Var olan uygulamalarınızda bu tür çağrılar varsa, bu işlevleri COBOL uygulamalarından kendi kitaplığınıza taşıyın; örneğin, `myMQ.so`. After moving these functions do not include the WebSphere MQ library `libmqmcbrt.o` when building the COBOL application for CICS.

   Ayrıca, COBOL uygulamanızın COBOL MQI çağrısı yapmazsa, `libmqmz_r` ile `cicsmkcobol`bağlantısını bağlamayın.

   Bu, Micro Focus COBOL dil yöntemi dosyasını oluşturur ve CICS Runtime COBOL kitaplığının UNIX and Linux systemssistemlerinde WebSphere MQ ' u aramasını sağlar.

   Not: `cicsmkcobol` komutunu yalnızca aşağıdaki ürünlerden birini kurduğunuzda çalıştırın:

   - Micro Focus COBOL ' in yeni sürümü veya
   - New version or release of TXSeries for Solaris
   - Desteklenen herhangi bir veritabanı ürününün yeni sürümü ya da yayın düzeyi (yalnızca COBOL işlemleri için)
   - New version or release of WebSphere MQ

2. Aşağıdaki ortam değişkenini dışa aktarın:

   ```
   COBCPY=MQ_INSTALLATION_PATH/inc  export COBCPY
   ```

3. Programı çevirerek, derleyin ve bağlayın:

```
cicstcl -l COBOL -e <yourprog>.ccp
```

CICS C programlarının hazırlanması

Standart CICS olanaklarını kullanarak CICS C programlarını oluşturun:

1. Aşağıdaki ortam değişkenlerinin ***bir*** öğesini dışa aktarın:
   - LDFLAGS = "-L*MQ_INSTALLATION_PATH*ALIND LIB -LMQM_R" LDFLAGS dışa aktar
   - USERLIB = "-L*MQ_INSTALLATION_PATH*add lib -lmqm_r" export USERLIB
2. Programı çevirerek, derleyin ve bağlayın:

```
cicstcl -l C amqscic0.ccs
```

CICS C örnek hareketi

Bir CICS WebSphere MQ işlemi için Örnek C kaynağı, AMQSCIC0.CCS. Hareket, varsayılan kuyruk yöneticisinde SYSTEM.SAMPLE.CICS.WORKQUEUE iletim kuyruğundan iletileri okur ve bunları, iletinin iletim üstbilgisinde yer alan bir kuyruk adıyla yerel kuyruğa yerleştirir. SYSTEM.SAMPLE.CICS.DLQ. Örnek MQSC komut kütüğünü (AMQSCIC0.TST , bu kuyrukları ve örnek giriş kuyruklarını yaratmanızı sağlar.

Uygulamanızı Windows sistemlerinde oluşturma

Windows sistem yayınları, yazdığınız programlardan yürütülebilir uygulamaların nasıl oluşturulacağını açıklar.

This topic describes the additional tasks, and the changes to the standard tasks, that you must perform when building WebSphere MQ for Pencereler applications to run under Pencereler systems. ActiveX, C, C + +, COBOL ve Visual Basic programlama dilleri desteklenmektedir. ActiveX programlarınızı hazırlamaya ilişkin bilgi için bkz. Bileşen Nesne Modeli Arabiriminin Kullanılması (WebSphere MQ Automation Classes for ActiveX). C++ programlarınızı hazırlamaya ilişkin bilgi için bkz. C++ kullanılması.

Pencereler için WebSphere MQ ile yürütülebilir bir uygulama oluşturmak için gerçekleştirmeniz gereken görevler, kaynak kodunuzun yazıldığı programlama diline göre değişiklik gösterir. In addition to coding the MQI calls in your source code, you must add the appropriate language statements to include the WebSphere MQ for Pencereler include files for the language that you are using. Bu dosyaların içeriğini kendinize tanıdık bir hale getiriniz. Tam açıklama için “IBM WebSphere MQ veri tanımlama dosyaları” sayfa 77 ' e bakın.

Pencerelerüzerinde 64 bit uygulamalar oluşturma

Hem 32 bitlik, hem de 64 bitlik uygulamalar IBM WebSphere MQ for Windows Version 7.5üzerinde desteklenir. IBM WebSphere MQ yürütülebilir dosyası ve kitaplık dosyaları hem 32 bit, hem de 64 bit biçimlerde sağlanır, birlikte çalıştığınız uygulamaya bağlı olarak uygun sürümü kullanın.

Yürütülür dosyalar ve kitaplıklar

IBM WebSphere MQ kitaplıklarının 32 bit ve 64 bit sürümleri aşağıdaki konumlarda sağlanır:

Çizelge 68. IBM WebSphere MQ kitaplıklarının yeri

Kitaplık sürümü	**Kitaplık dosyalarını içeren dizin**
32 bit	*MQ_INSTALLATION_PATH*\Tools\Lib
64 bit	*MQ_INSTALLATION_PATH*\Tools\Lib64

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

32 bit uygulamalar, geçiş işleminden sonra olağan şekilde çalışmaya devam eder. 32 bitlik dosyalar, ürünün önceki sürümleriyle aynı dizinde bulunur.

If you want to create 64-bit version you must ensure that your environment is configured to use the library files in *MQ_INSTALLATION_PATH*\Tools\Lib64. LIB ortam değişkeninin, 32 bit kitaplıkları içeren klasörü aramak üzere ayarlanmadığından emin olun.

Windows 'ta C programlarının hazırlanması

Tipik Windows ortamınızda çalışın; Windows için WebSphere MQ özel bir şey gerektirmez.

64 bit uygulamaların programlanması hakkında daha fazla bilgi için 64 bit platformlarda Coding standartlarıbaşlıklı konuya bakın.

- Programlarınızı, WebSphere MQtarafından sağlanan uygun kitaplıklarla bağlantılayın:

Kitaplık dosyası	Program/çıkış tipi
MQ_INSTALLATION_PATH\Tools\Lib\mqm.lib	32 bit C için sunucu
MQ_INSTALLATION_PATH\Tools\Lib\mqic.lib	32 bit C için istemci
MQ_INSTALLATION_PATH\Tools\Lib\mqicxa.lib	İşlem eşgüdümü olan 32 bitlik C için istemci
MQ_INSTALLATION_PATH\Tools\Lib64\mqm.lib	64 bit C için sunucu
MQ_INSTALLATION_PATH\Tools\Lib64\mqic.lib	64 bit C için istemci
MQ_INSTALLATION_PATH\Tools\Lib64\mqicxa.lib	İşlem eşgüdümü olan 64 bit C için istemci

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Aşağıdaki komut, amqsget0 örnek programının (Microsoft Visual C++ derleyicisi kullanılarak) derlenmesi için bir örnek verir.

32 bit uygulamalar için:

```
cl -MD amqsget0.c -Feamqsget.exe MQ_INSTALLATION_PATH\Tools\Lib\mqm.lib
```

64 bit uygulamalar için:

```
cl -MD amqsget0.c -Feamqsget.exe MQ_INSTALLATION_PATH\Tools\Lib64\mqm.lib
```

Not:

– Kurulabilir bir hizmet yazıyorsanız (ek bilgi için Yönetme başlıklı konuya bakın), mqmzf.lib kitaplığına bağlanmanız gerekir.

– IBM TXSeries Encinaya da BEA Tuxedo gibi XA uyumlu bir hareket yöneticisi tarafından dış eşgüdümü için bir uygulama üretiyorsanız, mqmxa.lib ya da mqmxa.lib kitaplığına bağlanmanız gerekir.

  - Bir CICS çıkışı yazıyorsanız, mqmcics4.lib kitaplığına bağlantı girin.
  - Diğer ürün kitaplıklarından önce WebSphere MQ kitaplıklarını bağlamanız gerekir.
- DLL ' ler belirttiğiniz yolda (PATH) yer almalıdır.
- If you use lowercase characters whenever possible, you can move from WebSphere MQ for Pencereler to WebSphere MQ on UNIX and Linux systems, where use of lowercase is necessary.

CICS ve Transaction Server programlarının hazırlanması

Bir CICS WebSphere MQ işlemi için Örnek C kaynağı, AMQSCIC0.CCS. Standart CICS olanaklarını kullanarak oluştururs un. Örneğin, Windows 2000 için TXSeries için:

1. Ortam değişkenini ayarlayın (bir satıra aşağıdaki kodu girin):

```
set CICS_IBMC_FLAGS=-IMQ_INSTALLATION_PATH\Tools\C\Include;
%CICS_IBMC_FLAGS%
```

2. USERLIB ortam değişkenini ayarlayın:

```
set USERLIB=MQM.LIB;%USERLIB%
```

3. Örnek programı çevirin, derleyin ve bağlayın:

```
cicstcl -l IBMC amqscic0.ccs
```

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Bu, *Transaction Server for Windows NT Application Programming Guide (CICS) V4*adlı kılavuzda açıklanmaktadır.

You can find more information about supporting CICS transactions in the Yönetme.

Preparing COBOL programs in Windows

Use this information to learn to prepare COBOL programs in Windows, and preparing CICS and Transaction Server programs.

1. 32 bit COBOL kopya kitapları şu dizine kurulur: *MQ_INSTALLATION_PATH*\Tools\cobol\CopyBook.
2. 64 bit COBOL kopya kitapları şu dizine kurulur: *MQ_INSTALLATION_PATH*\Tools\cobol\CopyBook64
3. Aşağıdaki örneklerde CopyBook to:

```
CopyBook
```

   32 bit uygulamalar için ve:

```
CopyBook64
```

   64 bit uygulamalar için.

MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

COBOL programlarını Windows sistemlerinde hazırlamak için, programınızı IBM WebSphere MQtarafından sağlanan aşağıdaki kitaplıklardan birine bağlayın:

Kitaplık dosyası	Program ya da çıkış tipi
MQ_INSTALLATION_PATH\Tools\Lib\mqmcbb	IBM COBOL için 32 bit sunucu
MQ_INSTALLATION_PATH\Tools\Lib\mqmcb	Micro Focus COBOL için 32 bit sunucu

Kitaplık dosyası	Program ya da çıkış tipi
MQ_INSTALLATION_PATH\Tools\Lib\mqiccbb	IBM COBOL için 32 bit istemci
MQ_INSTALLATION_PATH\Tools\Lib\mqiccb	Micro Focus COBOL için 32 bit istemci
MQ_INSTALLATION_PATH\Tools\Lib64\mqmcbb	IBM COBOL için 64 bit sunucu
MQ_INSTALLATION_PATH\Tools\Lib64\mqmcb	Micro Focus COBOL için 64 bit sunucu
MQ_INSTALLATION_PATH\Tools\Lib64\mqiccbb	IBM COBOL için 64 bit istemci
MQ_INSTALLATION_PATH\Tools\Lib64\mqiccb	Micro Focus COBOL için 64 bit istemci

Bir programı MQI istemcisi ortamında çalıştırırken, DOSCALLS kitaplığının, herhangi bir COBOL ya da IBM WebSphere MQ kitaplığından önce görünmesine dikkat edin.

You can use the IBM COBOL Set compiler or Micro Focus COBOL compiler depending on the program:

- amqi başlangıç programları, IBM COBOL Set derleyicisi için uygundur.
- amqm ' un başlangıç programları Micro Focus COBOL derleyicisi için uygundur ve
- amq0 başlangıç programları, her iki derleyici için uygundur.

IBM ve Micro Focus COBO

Var olan 32 bit IBM WebSphere MQ Micro Focus COBOL programlarını, mqmcbb ve mqiccbb kitaplıklarını değil, mqmcb.lib ya da mqiccb.libkullanarak yeniden bağlayın.

Örneğin, örnek program amq0put0, IBM VisualAge COBOL ' u kullanarak derlemek için:

1. SYSLIB ortam değişkenini, IBM WebSphere MQ VisualAge COBOL copybooks yolunu içerecek şekilde ayarlayın (tek bir satırda aşağıdaki kodu girin):

   ```
   set SYSLIB=MQ_INSTALLATION_PATH\
   Tools\Cobol\Copybook\VAcobol;%SYSLIB%
   ```

2. IBM WebSphere MQ sunucusunda kullanım için:

   ```
   cob2 amq0put0.cbl -qlib "MQ_INSTALLATION_PATH\
   Tools\Lib\mqmcbb.lib"
   ```

3. IBM WebSphere MQ istemcisinde kullanmak için:

   ```
   cob2 amq0put0.cbl -qlib "MQ_INSTALLATION_PATH\
   Tools\Lib\mqiccbb.lib"
   ```

   Not: CALLINT (SYSTEM) derleyici seçeneğini kullanmanız gerekse de, cob2için varsayılan değer budur.

To compile, for example, the sample program amq0put0, using Micro Focus COBOL:

1. COBCPY ortam değişkenini IBM WebSphere MQ COBOL copybook kitaplarını gösterecek şekilde ayarlayın (bir satıra aşağıdaki kodu girin):

   ```
   set COBCPY=MQ_INSTALLATION_PATH\
   Tools\Cobol\Copybook
   ```

2. Size bir nesne dosyası vermek için programı derleyin:

   ```
   cobol amq0put0 LITLINK
   ```

3. Nesne dosyasını çalıştırma zamanı sistemine bağlayın.

- LIB ortam değişkenini, derleyici COBOL kitaplıklarını işaret edecek şekilde ayarlayın.
- Nesne dosyasını IBM WebSphere MQ sunucusunda kullanılmak üzere bağlayın:

```
cbllink amq0put0.obj mqmcb.lib
```

- Ya da nesne dosyasını IBM WebSphere MQ istemcisinde kullanılmak üzere bağlayın:

```
cbllink amq0put0.obj mqiccb.lib
```

CICS ve Transaction Server programlarının hazırlanması

To compile and link a TXSeries for Windows NT, V5.1 program using IBM VisualAge COBOL:

1. Ortam değişkenini ayarlayın (bir satıra aşağıdaki kodu girin):

```
set CICS_IBMCOB_FLAGS=MQ_INSTALLATION_PATH\
Cobol\Copybook\VAcobol;%CICS_IBMCOB_FLAGS%
```

2. USERLIB ortam değişkenini ayarlayın:

```
set USERLIB=MQMCBB.LIB
```

3. Programınızı çevirin, derleyin ve bağlayın:

```
cicstcl -l IBMCOB myprog.ccp
```

Bu, *Transaction Server for Windows NT, V4 Application Programming* Guide adlı kılavuzda açıklanmaktadır.

Micro Focus COBOL kullanarak bir CICS for Windows V5 programını derlemek ve bağlamak için:

- INCLUDE değişkenini ayarlayın:

```
set
INCLUDE=<drive>:\<programname>\ibm\websphere\tools\c\include;
        <drive>:\opt\cics\include;%INCLUDE%
```

- COBCPY ortam değişkenini ayarlayın:

```
 setCOBCPY=<drive>:\<programname>\ibm\websphere\tools\cobol\copybook;
          <drive>:\opt\cics\include
```

- COBOL seçeneklerini ayarlayın:
  - set
  - COBOPTS=/LITLINK /NOTRUNC

ve aşağıdaki kodu çalıştırın:

```
cicstran cicsmq00.ccp
cobol cicsmq00.cbl /LITLINK /NOTRUNC
cbllink -D -Mcicsmq00 -Ocicsmq00.cbmfnt cicsmq00.obj
%CICSLIB%\cicsprCBMFNT.lib user32.lib msvcrt.lib kernel32.lib mqmcb.lib
```

Visual Basic Programlarinin Windows 'ta Hazırlanması

Pencerelerüzerinde Visual Basic programları kullanmayı göz önünde bulundurarak bu bilgileri kullanın.

Not: Visual Basic birim ktklerinin 64 bitlik srmleri sa§lanmamz.

Visual Basic programlarını Windows' a hazırlamak için:

1. Yeni bir proje yaratır.
2. Sağlanan modül dosyasını (CMQB.BAS, projeye.
3. Gereksinim duyarsanız, sağlanan diğer modül dosyalarını ekleyin:

CMQBB.BAS	MQAI desteği
CMQCFB.BAS	PCF desteği
CMQXB.BAS	Kanal çıkışları desteği
CMQPSB.BAS	Yayınla/abone ol

Visual Basic içinde MQCONNXAny çağrısının kullanılmasına ilişkin bilgi edinmek için “Visual Basic Kodlamasi” sayfa 83 belgesine bakın.

Proje kodunda MQI çağrıları yapmadan önce MQ_SETDEFAULTS yordamını çağırın. Bu yordam, MQI çağrılarının gerektirdiği varsayılan yapıları ayarlar.

*MqType*koşullu derleme değişkenini ayarlayarak projeyi derlemeden ya da çalıştırmadan önce bir WebSphere MQ sunucusu ya da istemcisi yaratıp yaratmayacağınızı belirtin. Bir sunucu için *MqType* , bir sunucu ya da 2 için bir Visual Basic projesinde aşağıdaki gibi bir istemci için 2 değeri ayarlayın:

1. Proje menüsünü seçin.
2. Select *Name* Properties (where *Name* is the name of the current project).
3. İletişim kutusunda Make (Make) etiketini seçin.
4. Koşullu Derleme Bağımsız Değişkenleri alanında, bu değeri bir sunucu için girin:

```
MqType=1
```

ya da bir istemci için:

```
MqType=2
```

SSPI güvenlik çıkışı

Pencereler için WebSphere MQ , hem WebSphere MQ MQI istemcisi hem de WebSphere MQ sunucusu için bir güvenlik çıkışı sağlar. Bu program, Security Services Programming Interface (SSPI) olanağını kullanarak WebSphere MQ kanalları için kimlik doğrulaması sağlayan bir kanal çıkış programıdır. SSPI, Pencereler sistemlerinin tümleşik güvenlik olanaklarını sağlar.

Güvenlik paketleri security.dll ya da secur32.dlliçinden yüklenir. Bu DLL ' ler işletim sisteminiz ile birlikte sağlanır.

NTLM kimlik doğrulama hizmetleri kullanılarak tek yönlü kimlik doğrulaması sağlanır. Kerberos kimlik doğrulama hizmetleri kullanılarak iki yönlü kimlik doğrulaması sağlanır.

Güvenlik çıkış programı kaynak ve nesne biçiminde sağlanır. Nesne kodunu olduğu gibi kullanabilir ya da kaynak kodu, kendi kullanıcı çıkış programlarınızı yaratmak için başlangıç noktası olarak kullanabilirsiniz.

Ayrıca bkz. “ Windows sistemlerinde SSPI güvenlik çıkışısının kullanılması” sayfa 162.

Güvenlik çıkışlarına giriş

Bir güvenlik çıkışı, iki güvenlik çıkış programı arasında güvenli bir bağlantı oluşturur; burada bir program, ileti kanalı aracısı (MCA) için bir programdır ve bir program MCA ' yı almak içindir.

Güvenli bağlantıyı başlatan program (örneğin, MCA oturumu kurulduktan sonra denetimi alan ilk program olan) *bağlam başlatıcısı*olarak bilinir. İş ortağı programı, *bağlam kabul edici*olarak bilinir.

Aşağıdaki çizelge, bağlam kullanıma hazırlayıcıları ve ilişkili bağlam kabul edenleri olan bazı kanal tiplerini gösterir.

Çizelge 69. Bağlam başlatıcıları ve bunların ilişkili bağlam kabul edilicileri

Bağlam Başlatıcısı	**Bağlam Kabul Edilicisi**
MQCHT_CLNTCONN	MQCHT_SVRCONN
MQCHT_RECEI	MQCHT_SENDER
MQCHT_CLAUSRCVR	MQCHT_CLUSSDR

Güvenlik çıkış programının iki giriş noktası vardır:

- **SCY_NTLM**

  Bu, tek yönlü kimlik doğrulaması sağlayan NTLM kimlik doğrulama hizmetlerini kullanır. NTLM, sunucuların, istemcilerinin kimliklerini doğrulamasına olanak sağlar. İstemcilerin bir sunucunun kimliğini doğrulamasına izin vermez ya da başka bir sunucunun kimliğini doğrulamak için bir sunucu doğrulamaz. NTLM kimlik doğrulaması, sunucuların orijinal olduğu varsayıldığı bir ağ ortamı için tasarlandı.

- **SCY_KERBEROS**

  Bu, Kerberos karşılıklı kimlik doğrulama hizmetlerini kullanır. Kerberos protokolü, bir ağ ortamındaki sunucuların gerçek olduğunu varsaymaz. Ağ bağlantısının her iki ucundaki taraflar, diğer tarafın kimliğini doğrulayabilir. Yani, sunucular istemcilerin ve diğer sunucuların kimliğini doğrulayabilir ve istemciler bir sunucunun kimliğini doğrulayabilir.

Güvenlik çıkıştan çıkılıyor

Bu konuda, SSPI kanal çıkış programlarının ne yaptığı açıklanmaktadır.

Sağlanan kanal çıkış programları, bir oturum kurulurken bir ortak sistemin tek yönlü ya da iki yönlü (karşılıklı) kimlik doğrulamasını sağlar. Belirli bir kanal için, her çıkış programının ilişkili bir *birincil kullanıcısı* (kullanıcı kimliği ile benzer şekilde, bkz. “WebSphere MQ erişim denetimi ve Windows asıl adları” sayfa 446) vardır. İki çıkış programı arasında bir bağlantı, iki birincil kullanıcı arasında bir ilişkilendirmeye sahip olur.

Temeldeki oturum oluşturulduktan sonra iki güvenlik çıkış programı arasında güvenli bir bağlantı (MCA ' nın gönderilmesi için biri ve alıcı MCA için bir program) kurulur. İşlem sırası aşağıdaki gibidir:

1. Her program, belirli bir birincil kullanıcıyla ilişkilendirilir; örneğin, açık bir oturum açma işleminin sonucu olarak.
2. Bağlam başlatıcı, güvenlik paketindeki ortakla güvenli bir bağlantı (Kerberos, adı ortak) ister ve bir simge alır (token1olarak adlandırılır). Simge, önceden kurulmuş olan temel oturumu ortak programa kullanarak gönderilir.
3. İş ortağı programı (bağlam kabul edici), bağlam başlatıcısının özgün olduğunu doğrulayan token1 ' i güvenlik paketine iletir. NTLM için şu anda bağlantı kurulur.
4. Kerberostarafından sağlanan güvenlik çıkışı (karşılıklı kimlik doğrulama için) için, güvenlik paketi, bağlam kabul cısının temeldeki oturumu kullanarak bağlam başlatıcısına döndürdüğü ikinci bir belirteç (token2olarak adlandırılır) oluşturur.
5. Bağlam başlatıcısı, bağlam kabul edilenin otantik olduğunu doğrulamak için token2 ' yi kullanır.
6. Bu aşamada, her iki uygulama da iş ortağının simginden memnunsa, güvenli (doğrulanmış) bağlantı kurulur.

WebSphere MQ erişim denetimi ve Windows asıl adları

WebSphere MQ ' un sağladığı erişim denetimi, kullanıcıya ve gruba dayanır. Pencereler ' in sağladığı kimlik doğrulaması, kullanıcı ve servicePrincipalAd (SPN) gibi birincil kullanıcıları temel alır. servicePrincipal(servicePrincipal) adı durumunda, tek bir kullanıcıyla ilişkilendirilmiş olan birçok öğe olabilir.

SSPI güvenlik çıkışı, kimlik doğrulaması için ilgili Windows asıl adlarını kullanır. Pencereler kimlik doğrulaması başarılı olursa, çıkış, erişim denetimi için Pencereler asıl adıyla WebSphere MQ ile ilişkili kullanıcı kimliğini iletir.

Kimlik doğrulaması için uygun olan Windows asıl adları, kullanılan kimlik doğrulama tipine bağlı olarak değişir.

- NTLM kimlik doğrulaması için, Context Initiator için Windows asıl adı, çalışmakta olan süreçle ilişkilendirilmiş olan kullanıcı kimliğidir. Bu kimlik doğrulaması bir yol olduğu için, Context Acceptor ile ilişkili asıl adın ilgisiz olduğu bir yöntem.
- Kerberos kimlik doğrulaması için, CLNTCONN kanallarında, Pencereler asıl adı, çalışmakta olan süreçle ilişkilendirilmiş olan kullanıcı kimliğidir. Otherwise, the Pencereler principal is the servicePrincipalName that is formed by adding the following prefix to the QueueManagerName.

```
ibmMQSeries/
```

Using lightweight directory access protocol services with WebSphere MQ for Pencereler

Bu konuda, bir dizin hizmetinin ne olduğu ve bir dizin erişim protokolü (DAP) tarafından kullanılan parça açıklanmıştır. It also explains how WebSphere MQ applications can use a lightweight directory access protocol (LDAP) directory using a sample program as a guide.

Not: Örnek program, LDAP ' ı önceden tanımış olan biri için tasarlanmıştır.

The following topics give more information about directory services, LDAP, and using LDAP with WebSphere MQ.

- "Dizin Hizmeti" sayfa 446
- "Lightweight Directory Access Protocol (LDAP)" sayfa 447
- "Using LDAP with WebSphere MQ" sayfa 447

Dizin Hizmeti

Dizin, nesnelerle ilgili, belirli bir nesnelerle ilgili bilgileri kolayca bulmanın kolay olduğu bir bilgi havuzundan oluşan bir havuzdur.

Ortak örnek, bilgilerin (adres ve telefon numarası) kişi ve şirketler hakkında depolandığı bir telefon dizinidir. Diğer bir örnek, e-posta adreslerinin ve isteğe bağlı olarak telefon numaraları gibi diğer bilgilerin kişiler için depolandığı bir e-posta sistemine ilişkin adres defteridir.

Bilgisayar sistemlerinde, dizinler, yazıcılar ya da paylaşılan diskler gibi bilgisayar kaynaklarına ilişkin bilgileri saklayabilir. Örneğin, en yakın renk yazıcısının nerede bulunduğunu öğrenmek için bir dizin kullanabilirsiniz. Bir WebSphere MQ uygulaması içinde, bir uygulama hizmeti (örneğin, hesap-alacak işlem gibi) arasındaki ilişkilendirmeyi ve bu hizmeti gerektiren iletiler için kullanılacak kuyruğu (büyük olasılıkla kuyruk adı ve anasistem kuyruk yöneticisi adı ile tespit edilir) sağlamak için bir dizin kullanılabilir.

Dizinler, istemci-sunucu sistemleri olarak gerçekleştirilir; burada dizin sunucusu, istemcilerden gelen tüm bilgileri ve yanıtlar isteklerini içerir. İstemciler, bilgileri doğrudan kullanıcıya sağlayan kullanıcı arabirimi programları ya da çalışmalarını tamamlamak için kaynakları bulmanız gereken uygulama programları da olabilir. Dizin Hizmeti, dizin sunucusunu, yönetim programlarını ve dizini yapılandırmak, güncellemek ve okumak için gerekli olan istemci kitaplıklarını ve programlarından oluşur.

Lightweight Directory Access Protocol (LDAP)

Novell Directory Services, DCE Cell Directory Service, Banyan StreetTalk, Windows Directory Services, X.500gibi birçok dizin hizmeti ve e-posta ürünleriyle ilişkili adres defteri hizmetleri vardır. X.500 , International Standards Organization (ISO) tarafından genel dizin hizmetleri için standart olarak önerildi. İletişim için bir OSI iletişim kuralı yığını gerektirir ve büyük oranda bu şekilde kullanımı büyük kuruluşlarla ve akademik kurumlarla sınırlı olmuştur. Bir X.500 dizin sunucusu, DAP ' yi (Dizin Erişimi Protokolü) kullanarak istemcileriyle iletişim kurar.

LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü), DAP ' ın basitleştirilmiş bir sürümü olarak yaratıldı. Uygulamak daha kolaydır, DAP 'in bazı Lesser-used özelliklerini çıkarır ve TCP/IP' nin üzerinde çalışır. Bu değişiklikler sonucunda, önceden kullanılan özel protokollerin birden çok olması nedeniyle, çoğu amaç için dizin erişimi protokolü olarak hızla benimsenir. LDAP istemcileri bir ağ geçidi üzerinden bir X.500 sunucusuna erişmeye devam edebilir (X.500 hala OSI iletişim kuralı yığınını gerektirir) ya da giderek artan X.500 uygulamaları genellikle DAP erişiminin yanı sıra LDAP için yerel destek de içerir.

LDAP dizinleri dağıtılabilir ve içeriklerine verimli bir şekilde erişilmesini sağlamak için eşlemeyi kullanabilir.

LDAP 'ın daha eksiksiz bir açıklaması için bkz. *LDAP' ı anlama*, bir IBM Redbooks yayınına bakın.

Using LDAP with WebSphere MQ

WebSphere MQ yapılandırmalarında, ileti ve iletim kuyruklarını tanımlayan bilgiler yerel olarak depolanır. This means that in a WebSphere MQ network the various definitions are distributed, with no central directory of this information being available for browsing. WebSphere MQ uygulamaları arasında uzak ileti sistemi, genellikle uzak kuyrukların yerel tanımlamalarının kullanımıyla elde edilir. Uygulama ilk olarak, uzak kuyruğun yerel tanımlamasında belirtilen adı kullanarak bir MQOPEN çağrısı yayınlar. İletiyi uzak kuyruğa koymak için, uygulama MQPUT çağrısından döndürülen tanıtıcıyı belirterek, MQPUT ' un çağrısını verir. Uzak kuyruk tanımlaması, hedef kuyruğunun adını, hedef kuyruk yöneticisini ve isteğe bağlı olarak bir iletim kuyruğunu sağlar. Bu tekniğin içinde uygulamanın, yerel kuyruk tanımında belirlenen adı çalıştırma zamanı bilmesi gerekir.

Önceki bir değişkendeki bir varyasyon, uzak kuyrukların yerel tanımlamalarının kullanımını önler. Uygulama, MQAPER ' ın bir parçası olarak uzak kuyruk yöneticisi adını içeren tam hedef kuyruk adını belirleyebilir. Bu nedenle, uygulamanın yürütme sırasında bu iki adı bilmesi gerekir. Yerel kuyruk yöneticisi, yerel kuyruk tanımlamasıyla ve uygun bir şekilde adlandırılmış (ya da varsayılan) iletim kuyruğu ve hedefe teslim eden ilişkili bir kanalla doğru olarak yapılandırılmış olmalıdır.

Hem kaynak hem de hedef kuyruk yöneticilerinin aynı kümenin üyesi olarak tanımlandığı durumlarda, önceki iki senaryoya ilişkin iletim kuyruğu ve kanal açıları yoksayılabilir. Hedef iletim kuyruğu bir küme kuyruğunysa, uzak kuyruğun yerel tanımlaması da gerekmez. Ancak, açıklanan önceki vakalara benzer şekilde, uygulamanın hedef kuyruğun adını yine de bilmesi gerekir.

Bu uygulama bağımlılığını kuyruk adlarına (ya da kuyruk ve kuyruk yöneticisi adlarının birleşimine) kaldırmak için bir dizin hizmeti kullanılabilir. Uygulama ölçütleri ile WebSphere MQ nesne adları arasındaki eşleme, bir dizinde tutulabilir ve dinamik olarak güncellenebilir ve uygulamalardan bağımsız olarak güncellenebilir. Bir iletiyi göndermek isteyen WebSphere MQ uygulaması, dizine uygulama tabanlı ölçütler kullanarak sorgular; örneğin: service_name = "accounts receivable", ilgili WebSphere MQ nesne adlarını alır ve sonra bu döndürülen değerleri MQOPEN çağrısında kullanır.

Dizin kullanımının başka bir örneği, çok sayıda küçük depoya ya da ofislere sahip bir şirket için, WebSphere MQ MQI istemcileri, daha büyük ofislerde bulunan WebSphere MQ sunucularına ileti göndermek için kullanılabilir. İstemciler, ileti gönderdikleri her sunucu için anasistem makinelerinin, MQI kanalının ve kuyruk adının adını bilmeye gerek duyarlar. Ara sıra, bir WebSphere MQ sunucusunu başka bir makineye taşımak gerekebilir; sunucuyla iletişim kuran her istemcinin değişikliği bilmesi gerekir. Bir LDAP dizin hizmeti, anasistem makinelerinin adlarını (ve kanal ve kuyruk adlarını) saklamak için kullanılabilir ve istemci programları bir sunucuya ileti göndermek istediklerinde bilgileri dizinden alabilirler. Bu durumda, bir anasistem adı (ya da kanal ya da kuyruk adı) değiştirildiyse, yalnızca dizinin güncellenmesi gerekir.

Bir uygulama iletisi için birden çok hedef bir dizinde saklanabilir; seçilen kişi, kullanılabilirliği ya da yük paylaşımı konusunda dikkate alınması gereken hususlara bağımlı olarak seçilir.

WebSphere MQ , SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) ile kullanılmak üzere kimlik doğrulama bilgilerini saklamak için bir LDAP dizini de kullanabilir. Java için WebSphere MQ sınıfları da bilgileri bir LDAP dizininde saklayabilir.

LDAP örnek programı

Örnek program, LDAP ' ı bilen ve büyük olasılıkla bunu kullanan bir kullanıcı için tasarlanmıştır. WebSphere MQ uygulamalarının bir LDAP dizinini nasıl kullanabileceğini göstermek amaçlanır.

Örnek programın oluşturulması

Bu program yalnızca TCP/IP 'yi kullanarak Windows ' da oluşturulmuştur ve sınanmıştır. “Windows 'ta C programlarının hazırlanması” sayfa 440içinde sözü edilen genel konuların yanı sıra aşağıdaki noktalara dikkat edin:

- Bu program bir istemci programı olarak çalışmak üzere tasarlanmıştır, bu nedenle MQIC.LIB kitaplığı.
- WebSphere MQ üstbilgi kütükleri ve kitaplıklarının yanı sıra, bu programın LDAP istemcisi üstbilgi kütükleri ve kitaplıkları kullanılarak oluşturulması gerekir.

  Örneğin, IBM eNetwork istemcisinin kullanılması, programı LIBLDAPSTATICE.LIB ve LIBLBERSTATICSSL.LIB kitaplıkları.

Dizinin yapılandırılması

Örnek program çalıştırılabilmesi için, örnek verilerle bir LDAP Dizin Sunucusu yapılandırılmalıdır.

`tools\c\samples` dizinindeki MQuser.ldifdosyası, LDIF ' te (LDAP Data Interchange Format) bazı örnek verileri içerir. Bu dosyayı gereksinimlerinize uyacak şekilde düzenleyebilirsiniz. Bu, üç ofisi oluşturan bir Taşıma Departmanına sahip MQuser adlı hayali bir şirketin verilerini içerir. Bu ofislerin her birinin, bir WebSphere MQ Server çalıştıran bir makinesi vardır.

Alt sınır olarak, WebSphere MQ sunucularını çalıştıran makinelerin anasistem adlarını içeren üç satırı düzenlemeniz gerekir: satır 18, 27 ve 36:

```
host: LondonHost
  ...
host: SydneyHost
  ...
host: WashingtonHost
```

`LondonHost`, `SydneyHost`ve `WashingtonHost` seçeneklerini, WebSphere MQ sunucularını çalıştıran makinelerinizin üçünün adlarına değiştirmeniz gerekir. Ayrıca, isterseniz kanal ve kuyruk adlarını da değiştirebilirsiniz (örneğin, sistem varsayılanlarının adlarını kullanır). Ayrıca, örnek verilerdeki ofislerin sayısını artırmak ya da azaltmak isteyebilirsiniz.

IBM Tivoli dizin sunucusunun yapılandırılması

Dizinin kurulmasıyla ilgili bilgi için IBM Tivoli Directory Server (ITDS) Administrator's Guide adlı kılavuza bakın. `Installing and Configuring Server`başlıklı konuda, `Installing Server` ve `Basic Server Configuration`kısımlarıyla çalışın. Gerekirse, arabirimin nasıl işlediğini öğrenmek için `Administrator Interface` konusunu okuyun.

`Configuring - How Do I`başlıklı konuda, yöneticiyi başlatmak için yönergeleri izleyin, daha sonra `Configure Database` bölümü aracılığıyla çalışın ve varsayılan bir veritabanı oluşturun. `Configure replica` bölümünü atlayın ve `Work with Suffixes`bölümünü kullanarak o=`MQusers`onekini ekleyin.

Veritabanına giriş eklemeden önce, bazı öznitelik tanımlamaları ve bir nesne sınıfı tanımlaması ekleyerek, dizin şemasını genişletmeniz gerekir. This is described in the IBM Tivoli Directory Server Administrator's Guide in the chapter `Reference Information` under the section `Directory Schema`. Bu konuda size yardımcı olmak üzere iki örnek dosya yer alır. `mq.at.conf` dosyası, `?etc?slapd.at.conf`dosyasına

eklemeniz gereken öznitelik tanımlamalarını içerir. `slapd.at.conf` dosyasını düzenleyerek ve bir satır ekleyerek örnek dosyayı ekleyerek bunu yapın:

```
include <pathname>/mq.at.conf
```

Diğer bir seçenek olarak, `slapd.at.conf` dosyasını düzenleyebilir ve örnek dosyanın içeriğini doğrudan ona ekleyebilirsiniz; yani, satırları ekleyin:

```
# MQ attribute definitions
attribute mqChannel           ces   mqChannel        1000   normal
attribute mqQueueManager      ces   mqQueueManager   1000   normal
attribute mqQueue             ces   mqQueue          1000   normal
attribute mqPort              cis   mqPort           64     normal
```

Similarly for the objectclass definition, you can either include the sample file by editing `etc?slapd.oc.conf` and add the line:

```
include <pathname>/mq.oc.conf
```

ya da örnek dosyanın içeriğini doğrudan `slapd.oc.conf,` ' e eklemek için, aşağıdaki satırları ekleyin:

```
# MQ object classdefinition
objectclass mqApplication
      requires
              objectClass,
              cn,
              host,
              mqChannel,
              mqQueue
      allows
              mqQueueManager,
              mqPort,
              description,
              l,
              ou,
              seeAlso
```

Şimdi dizin sunucusunu (Denetim, Sunucu, Başlatma) başlatabilir ve örnek girişleri bu girişlere ekleyebilirsiniz. Örnek girdileri eklemek için, denetimcinin Girdiler Ekle sayfasına gidin, örnek dosyanın tam yol adını `MQuser.ldif` yazın ve Gönder düğmesini tıklatın.

Dizin sunucusu şimdi çalışıyor ve örnek programı çalıştırmak için uygun verilerle yükleniyor.

Netscape dizin sunucusunun yapılandırılması

Netscape Server Administration sayfasını kullanarak **Create New Netscape Directory Server**(Yeni Netscape Dizin Sunucusu Oluştur) seçeneğini tıklatın.

Şimdi, yapılandırma bilgilerini içeren bir form ile sunulabilmelisiniz. Dizin Sonekini **o = MQuser** olarak değiştirin ve Unrestricted User (Sınırsız Kullanıcı) için bir parola ekleyin. Ayrıca, kuruluşunuza uyacak diğer bilgileri de değiştirebilirsiniz. **Tamam**düğmesini tıklatın; dizin başarıyla yaratılmalıdır. **Sunucu Denetimi 'ye Geri Dön** ' ü tıklatın ve dizin sunucusunu başlatın. Yeni dizin için Dizin Sunucusu Denetim sunucusunu başlatmak için dizin adını tıklatın.

Veritabanına giriş eklemeden önce, bazı öznitelik tanımlamaları ve bir nesne sınıfı tanımlaması ekleyerek dizin şemasını genişletin. Directory Server (Dizin Sunucusu) sayfasının **Schema** (Şema) etiketini tıklatın. Şimdi, yeni öznitelikler eklemenizi sağlayan bir form sunuyorsunuz. Aşağıdaki öznitelikleri ekleyin (tüm bunlar için öznitelik OID ' yi boş bırakın):

```
Attribute Name              Syntax
--------------              ------
mqChannel                   Case Exact String
mqQueueManager              Case Exact String
```

```
mqQueue                     Case Exact String
mqPort                      Integer
```

Yan panelde **Create ObjectClass** (Nesne Sınıfı Oluştur) seçeneğini tıklatarak yeni bir objectClass (nesne sınıfı) ekleyin. ObjectClass Adı olarak **mqApplication** girin, üst ObjectClass olarak **applicationProcess** seçeneğini belirleyin ve **ObjectClass OID** alanını boş bırakın. Şimdi objectClass' a bazı öznitelikler ekleyin. Gerekli öznitelikler olarak **anasistem**, **mqChannel**ve **mqQueue** öğelerini seçin ve izin verilen öznitelikler olarak **mqQueueManager** ve **mqPort** öğelerini seçin. objectClassöğesini yaratmak için **Create New ObjectClass** düğmesine basın.

Örnek verileri eklemek için, **Veritabanı Yönetimi** etiketini tıklatın ve yan panodan **Girdiler Ekle** seçeneğini belirleyin. Enter the path name of the sample data file <pathname>\MQuser.ldif, enter the password, and click **Tamam**.

Örnek program yetkisiz bir kullanıcı olarak çalışır ve varsayılan olarak Netscape Directory yetkisi olmayan kullanıcıların dizinde arama yapmalarına izin vermez. Bunu, **Erişim Denetimi** sekmesini tıklatarak değiştirin. Sınırsız Kullanıcı için parolayı girin ve dizine ilişkin erişim denetimi girdilerinde yüklemek için **Tamam** düğmesini tıklatın. Bunlar şu anda boş olmalıdır. Yeni bir erişim denetimi girdisi oluşturmak için **Yeni ACI** düğmesine basın. Görüntülenen giriş kutusunda, **Reddet** ' i tıklatın (altı çizili) ve sonuç iletişim kutusunda, bunu **İzin Ver**olarak değiştirin. Örneğin, **MQuser-access**gibi bir ad ekleyin ve **o = MQuser**seçeneğini belirlemek için **bir sonek seçin**seçeneğini tıklatın. Hedef olarak **o = MQuser** değerini girin, Unrestricted User için parolayı girin ve **Submit**(Gönder) düğmesini tıklatın.

Dizin sunucusu şimdi çalışıyor ve örnek programı çalıştırmak için uygun verilerle yükleniyor.

Örnek programın çalıştırılması

Şimdi, örnek verilerle çalışan ve doldurulan bir LDAP Directory Server 'a sahip olmasınız. Veriler, bunların tümünün WebSphere MQ sunucularını çalıştırıyor olması gereken üç anasistem makinelerini belirtir. Varsayılan kuyruk yöneticisinin her makinede çalıştırıldığından emin olun (örnek verileri farklı bir kuyruk yöneticisi belirtebilmek için değiştirmediyseniz).

Ayrıca, her makinede WebSphere MQ dinleyici programını başlatın; örnek, TCP/IP ' yi varsayılan WebSphere MQ kapı numarasıyla birlikte kullanır; böylece, aşağıdaki komutla dinleyiciye başlayabilirsiniz:

```
runmqlsr -t tcp
```

Örneği sınamak için, her bir WebSphere MQ sunucusuna gelen iletileri okumak için bir program çalıştırmak da isteyebilirsiniz; örneğin, amqstrg örnek programını kullanabilirsiniz:

```
amqstrg SYSTEM.DEFAULT.LOCAL.QUEUE
```

Örnek program, bir zorunlu ve iki isteğe bağlı olmak üzere üç ortam değişkenini kullanır. Gerekli değişken, dizin araması için temel ayırt edici adı belirten LDAP_BASEDN ' dir. Örnek verilerle çalışmak için, örneğin, Windows sistem tipindeki bir komut isteminde bu değeri ou=Transport, o=MQuserolarak ayarlayın:

```
set LDAP_BASEDN=ou=Transport, o=MQuser
```

İsteğe bağlı değişkenler LDAP_HOST ve LDAP_VERSION değişkenleridir. LDAP_HOST değişkeni, LDAP sunucusunun çalıştığı anasistemin adını belirtir; varsayılan değer olarak, yerel anasistemin adı belirtilir. LDAP_VERSION değişkeni, kullanılacak LDAP protokolünün sürümünü belirtir ve 2 ya da 3 olabilir. Çoğu LDAP sunucusu, protokolün 3. sürümünü destekliyor; bunların tümü de eski sürüm 2 'nin desteklenmesini destekler. Bu örnek, protokolün her iki sürümüyle de aynı şekilde çalışır ve belirtilmediyse, varsayılan olarak sürüm 2 'ye ayarlanır.

You can now run the sample by typing the program name followed by the name of the WebSphere MQ application that you want to send messages to, in the case of the sample data the application names are London, Sydney, and Washington. Örneğin, Londra uygulamasına ileti göndermek için:

```
amqsldpc London
```

Program WebSphere MQ sunucusuna bağlanmadıysa, uygun bir hata iletisi görüntülenir. Başarılı bir şekilde bağlantı kesilirse, yazdığınız her satır (< return> ya da < enter> tarafından sonlandırılmış) her bir satır ayrı bir ileti olarak gönderilir, boş bir satır programı sona erdirir.

Program tasarımı

Programın iki ayrı bölümü vardır: İlk kısım, bir LDAP dizin sunucusunu sorgulamak için ortam değişkenlerini ve komut satırı değerini kullanır; ikinci kısım, dizinden döndürülen bilgileri kullanarak WebSphere MQ bağlantısını kurar ve iletileri gönderir.

Programın ilk bölümünde kullanılan LDAP çağrıları, LDAP sürüm 2 ya da 3 'in kullanılıp kullanılmadığına bağlı olarak farklılık gösterir ve LDAP istemci kitaplıklarıyla birlikte gönderilen belgeler ayrıntılı olarak açıklanmıştır. Bu bölüm kısa bir açıklama sağlar.

Programın ilk bölümü, doğru çağrıldığını ve ortam değişkenlerini okuduğunu denetler. Daha sonra, belirtilen anasistemdeki LDAP dizin sunucusuyla bağlantı kurar:

```
if (ldapVersion == LDAP_VERSION3)
{
  if ((ld = ldap_init(ldapHost, LDAP_PORT)) == NULL)
     ...
}
else
{
  if ((ld = ldap_open(ldapHost, LDAP_PORT)) == NULL )
     ...
}
```

Bir bağlantı kurulduğunda, program "ldap_set_option" çağrısıyla sunucudaki bazı seçenekleri ayarlar ve daha sonra, sunucuya bağlanarak kendisini sunucuya doğrulayan bir şekilde sunucu ile gerçekleştirir:

```
if (ldapVersion == LDAP_VERSION3)
{
  if (ldap_simple_bind_s(ld, bindDN, password) != LDAP_SUCCESS)
     ...
}
else
{
  if (ldap_bind_s(ld, bindDN, password, LDAP_AUTH_SIMPLE) !=
      LDAP_SUCCESS)
     ...
}
```

`bindDN` ve `password` örnek programında NULL (boş) olarak ayarlıdır. Bu, programın kendisini anonim bir kullanıcı olarak doğruladığı anlamına gelir; yani, özel erişim haklarına sahip değildir ve yalnızca genel kullanıma açık bilgilere erişebilir. Uygulamada, çoğu kuruluş, yalnızca yetkili kullanıcıların erişebilmesi için dizinlerde sakladıkları bilgilere erişimi kısıtlıyor.

The first parameter to the bind call `ld` is a handle that is used to identify this particular LDAP session throughout the rest of the program. Kimlik doğrulandıktan sonra, program, uygulama adıyla eşleşen girdiler için dizinde arama yapar:

```
rc = ldap_search_s(ld,                /* LDAP Handle             */
                   baseDN,            /* base distinguished name */
                   LDAP_SCOPE_ONELEVEL, /* one-level search       */
                   filterPattern,     /* filter search pattern   */
                   attrs,             /* attributes required     */
                   FALSE,             /* NOT attributes only     */
                   &ldapResult);      /* search result           */
```

Bu, doğrudan sonuçları döndüren, sunucuya basit bir zamanuyumlu çağrı. Karmaşık sorgulara daha uygun olan ya da çok sayıda sonuç beklendiğinde başka arama türleri de vardır. The first parameter to the search is the handle `ld` that identifies the session. İkinci parametre, aramanın başlatılacağı dizinde yer alan temel ayırt edici addır ve üçüncü parametre, arama kapsamı, yani başlangıç noktasına göre girdilerin arandığı temel adıdır. Bu iki parametre, dizindeki hangi girişlerin arandığı tanımlanır. Bir sonraki parametre olan `filterPattern` , aradığımız şeyi belirtir. `attrs` parametresi, nesne bulunduğunda

nesneden geri almak istediğimiz öznitelikleri listeler. Sonraki öznitelik, yalnızca öznitelikleri mi, yoksa bunların değerlerini mi istediğimizi, bunun FALSE olarak ayarlanmasını, öznitelik değerlerinin istediğimizi ifade eder. Sonucu döndürmek için son parametre kullanılır.

Sonuç, her biri belirtilen özniteliklere ve değerlerine sahip birçok dizin girdisi içerebilir. Sonuçtan istediğimiz değerleri çıkarmamız gerekiyor. Bu örnek programda, yalnızca bir girişin bulunabilmesi için, sonuçtaki ilk girişe bakmamız gerekiyor:

```
ldapEntry = ldap_first_entry(ld, ldapResult);
```

Bu çağrı, ilk girdiyi temsil eden bir tanıtıcı döndürür ve girişten tüm öznitelikleri çıkarmak için bir for döngüsü ayarlarız:

```
for (attribute = ldap_first_attribute(ld, ldapEntry, &ber);
     attribute != NULL;
     attribute = ldap_next_attribute(ld, ldapEntry, ber ))
{
```

Bu özniteliklerin her biri için, bu özniteliklerle ilişkili değerleri çıkarırız. Yine her öznitelik için tek bir değer bekliyoruz, bu nedenle yalnızca ilk değeri kullanırız; hangi özniteliğe sahip olduğumuzu ve değeri uygun program değişkeninde sakladığımızı belirliyoruz:

```
values = ldap_get_values(ld, ldapEntry, attribute);
if (values != NULL && values[0] != NULL)
{
  if (stricmp(attribute, MQ_HOST_ATTR) == 0)
  {
    mqHost = strdup(values[0]);
    ...
```

Sonunda bellek boşaltarak (ldap_value_free, ldap_memfree, ldap_msgfree) serbest bırakıyoruz ve sunucudan *unbinding* ile oturumu kapatıyoruz.

```
ldap_unbind(ld);
```

We check that we have found all the WebSphere MQ values that we need from the directory, and if so we call sendMessages() to connect to the WebSphere MQ server and send the WebSphere MQ messages.

Örnek programın ikinci bölümü, tüm WebSphere MQ çağrılarını içeren sendMessages(sendMessages) yordamsıdır. Bu, amqsput0 örnek programında modellenmiş, program için parametrelerin genişletildiği farklar ve MQCONN çağrısı yerine MQCONNX kullanılır.

IBM WebSphere MQ Telemetryiçin uygulama geliştirilmesi

Telemetri uygulamaları, algılama ve kontrol cihazlarını internet üzerinde ve işletmelerde bulunan diğer bilgi kaynaklarıyla bütünleştirir.

Tasarım örüntülerini, çalışma örneklerini, örnek programları, programlama kavramlarını ve başvuru bilgilerini kullanarak IBM WebSphere MQ Telemetry için uygulamalar geliştirin. Birçok küçük aygıtı IBM WebSphere MQürününe bağlamayı kolaylaştırmak için aygıtlar için IBM WebSphere MQ Telemetry yardımcı programını kullanın.

İlgili kavramlar

WebSphere MQ Telemetry

İzleme ve denetim için telemetri kavramları ve senaryoları

İlgili görevler

WebSphere MQ Telemetrykuruluyor

WebSphere MQ TelemetryYönetimi

WebSphere MQ Telemetryiçin sorun giderme

İlgili başvurular
WebSphere MQ Telemetry Başvurusu

IBM WebSphere MQ Telemetry Örnek programlar

Örnek komut dosyaları, MQ Telemetry Transport v3 Client uygulamasının temel kullanımını göstermek için sağlanmıştır. Bir iletiyi yayınlamak ve bir konuya abone olmak için komut dosyalarını kullanın.

Başlamadan önce

Örnek programları çalıştırmak için telemetri (MQXR) hizmetini başlatın.

Kullanıcı kimliği, mqm kullanıcı grubunun bir üyesi olmalıdır.

Bir yayınlama ve abone olma işlemi gerçekleştirmek için önce SampleMQM komut dosyasını önce MQTTV3Sample komut dosyasını çalıştırın. SampleMQM komut dosyası tarafından yaratılan kuyruk yöneticisini silmek için CleanupMQM örnek komut dosyasını çalıştırın.

As the SampleMQM script creates and uses a queue manager called QM1, do not run unaltered on a system with a QM1 queue manager. Yapılan değişiklikler, var olan kuyruk yöneticisinin yapılanışlarına ilişkin sonuçları olabilir.

Bu görev hakkında

- SampleMQM uygulaması, QM1adı verilen bir telemetri etkin kuyruk yöneticisi yaratır ve başlatır. Komut dosyası ayrıca, QM1için varsayılan bir iletim kuyruğu ayarlar ve 1883 numaralı bağlantı noktasında varsayılan bir kanal dinleme işlemi oluşturur ve başlatır. Bu kanal, bu kanala bağlı istemcilerin kimlik doğrulamasını gerçekleştirmez. Kanalda ileti kanalı aracısı kullanıcı kimliği (MCAUSER) özniteliği var, Windows sistemlerinde 'guest' olarak ya da Linux sistemlerinde 'kimse' olarak ayarlanır. Kanala bağlı olan istemciler, çalışmakta olduğu sisteme bağlı olarak, kullanıcının 'konuk' ya da 'kimse' kullanıcısı gibi davranılır. Komut dosyası, Windows sistemlerinde 'guest' ve Linux sistemlerinde 'kimse' olarak yayınlanabilir ve QM1ile ilgili herhangi bir konuya abone olabilir.
- MQTTV3Sample uygulaması aşağıdaki konumda tutulur;
  - Windows *MQ_INSTALLATION_PATH*\mqxr\samples' da

    Burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQ ' un kurulu olduğu konumdur.
  - Linux *MQ_INSTALLATION_PATH*/mqxr/samples' da

  MQTTV3Sample uygulaması bir yayınlayıcı olarak işlev görerek, sunucudaki bir konuya tek bir ileti gönderir. Ayrıca, sunucudan gelen iletileri dinleyerek bir abone olarak da işlev görür.
- CleanupMQM örnek komut dosyası, SampleMQM komut dosyası tarafından yaratılan QM1 dosyasını sona erdirir ve siler. SampleMQM komut dosyasını yeniden çalıştırmak ya da QM1dosyasını kaldırmak istiyorsanız, CleanupMQM örnek komut dosyasını kullanın.

Yordam

1. SampleMQM komut dosyasını çalıştırmak için bir komut satırına aşağıdaki komutu yazın.
   - Windowsüzerinde, SampleMQM komut dosyasını çalıştırmak için komut aşağıdaki gibidir:

     ```
     MQ_INSTALLATION_PATH\mqxr\samples\SampleMQM.bat
     ```
   - AIX ve Linuxüzerinde, SampleMQM komut dosyasını çalıştırmak için kullanılan komut aşağıdaki gibidir:

     ```
     MQ_INSTALLATION_PATH/mqxr/samples/SampleMQM.sh
     ```

   Burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQ ' un kurulu olduğu konumdur.

   MQXR_SAMPLE_QM adlı bir kuyruk yöneticisi yaratıldı.

2. MQTTV3Sample komut dosyasının ilk bölümünü çalıştırmak için aşağıdaki komutu yazın.
   - Windows' ta, bir komut satırına aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH\mqxr\samples\RunMQTTV3Sample.bat -a subscribe
     ```

   - AIX ve Linuxüzerinde, bir kabuk penceresinde aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH/mqxr/samples/RunMQTTV3Sample.sh -a subscribe
     ```

3. MQTTV3Sample komut dosyasının ikinci bölümünü çalıştırmak için aşağıdaki komutu yazın.
   - Windows' ta, başka bir komut satırına aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH\mqxr\samples\RunMQTTV3Sample.bat -m "Hello from an MQTT v3
     application"
     ```

   - AIX ve Linuxüzerinde, başka bir kabuk penceresinde aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH/mqxr/samples/RunMQTTV3Sample.sh -m "Hello from an MQTT v3 application"
     ```

4. SampleMQM komut kütüğü tarafından yaratılan kuyruk yöneticisini kaldırmak için, aşağıdaki komutu kullanarak CleanupMQM komut dosyasını çalıştırabilirsiniz;
   - Windows' ta aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH\mqxr\samples\CleanupMQM.bat
     ```

   - AIX ve Linux ' da başka bir kabuk penceresinde aşağıdaki komutu yazın;

     ```
     MQ_INSTALLATION_PATH/mqxr/samples/CleanupMQM.sh
     ```

Sonuçlar

İkinci pencereye yazdığınız `Hello from an MQTT v3 application` iletisi, uygulama tarafından yayınlanır ve ilk pencerede uygulama tarafından alınır. İlk pencerede uygulama ekranda gösterecektir.

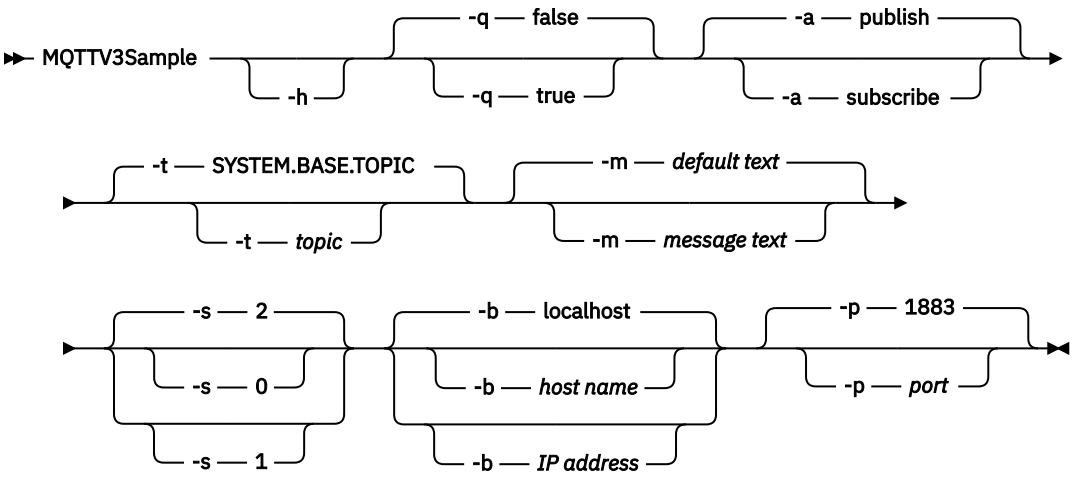
MQTTV3Sample programı

MQTTV3Sample programına ilişkin örnek sözdizimi ve parametrelere ilişkin başvuru bilgileri.

Amaç

MQTTV3Sample programı, bir iletiyi yayınlamak ve bir konuya abone olmak için kullanılabilir.

MQTTV3Sample syntax



Parametreler

-h
Bu yardım metnini yazdır ve çık

-q
False (yanlış) kipinin varsayılan kipini kullanmak yerine sessiz kipi ayarlayın.

-a
Yayınlamanın varsayılan eylemini varsaymak yerine yayınlama ya da abone olma işlemini ayarlayın.

-t
Varsayılan konuya abone olmak ya da varsayılan konuya abone olmak yerine, konuyu yayınlayın ya da konuya abone olun

-m
İleti metnini, varsayılan yayın metnini göndermek yerine, "Bir MQTT v3 uygulamasından merhaba" yayınına göndermek yerine yayınlayın.

-s
Varsayılan QoS, 2 değerini kullanmak yerine QoS değerini belirleyin.

-b
Varsayılan anasistem adına (localhost) bağlanmak yerine bu anasistem adına ya da IP adresine bağlanın.

-p
1883 varsayılan değerini kullanmak yerine bu kapıyı kullanın.

MQTTV3Sample programını çalıştırın.

Windows' ta bir konuya abone olmak için şu komutu kullanın:

```
runMQTTV3Sample -a subscribe
```

Pencereler' ta bir iletiyi yayınlamak için şu komutu kullanın:

```
runMQTTV3Sample
```

Sağlanan örnek komut dosyalarının çalıştırılmasıyla ilgili daha fazla bilgi için bkz. “IBM WebSphere MQ Telemetry Örnek programlar” sayfa 453.

Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması

Bir MQTT istemci uygulaması yaratma adımları, öğretici program modasında açıklanmıştır. Her kod satırı açıklanıyor. Görevin sonunda, bir MQTT yayınlayıcısı yaratmış bulunmuyorsunuz. WebSphere MQ Explorer olanağını kullanarak yayınlara göz atabilirsiniz.

Başlamadan önce

WebSphere MQ Telemetry özelliğini, IBM WebSphere MQ Version 7.1 ya da daha sonraki bir sürümü kurulu olan bir sunucuya kurun.

İstemci uygulaması, IBM WebSphere MQ Telemetry Software Development Toolkit (SDK) içinde `com.ibm.mq.micro.client.mqttv3` paketini kullanır. SDK, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır. The client connects to the IBM WebSphere MQ Telemetry feature to exchange messages with IBM WebSphere MQ.

IBM WebSphere MQ Telemetry' i yönetmek için IBM WebSphere MQ Explorer Version 7.1 ile ilgili telemetri güncellemelerini de kurmalısınız. Güncellemeler, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır.

Java SE üzerinde çalışan bir MQTT istemcisi, Java SE ' nin 6.0 sürümünü ya da üstünü gerektirir. IBM Java SE v6.0 , IBM WebSphere MQ Version 7.1 kuruluşunun bir parçasıdır. Şu konumda bulunur: *WebSphere MQ installation directory*\java\jre

Bu görev hakkında

Örnek, bir yayınlama uygulamasıdır, PubSync. PubSync , `MQTT Examples`başlıklı konuda `Hello World` yayınlanıyor ve yayının kuyruk yöneticisine teslim edildiğine ilişkin onay bekliyor.

By setting up a durable subscription to `MQTT Examples` you can check that the application works.

Yordam, istemciyi geliştirmek, oluşturmak ve çalıştırmak için Eclipse ' i kullanır. You can download Eclipse from the Eclipse project website at www.eclipse.org.

Uygulamayı yaratmak için, Java dosyalarını yaratabilir ve komut satırını kullanarak bunları derleyebilir ve çalıştırabilirsiniz.

Yeni bir dizinde `.\com\ibm\mq\id`dizin yolunu yaratın. İki Java dosyası (`Example.java` ve PubSync`.java`) oluşturun. Kodu “Örnek kod” sayfa 459 ' den Java dosyalarına kopyalayın.

Komutu kullanarak Java kodunu derleyin,

```
javac -cp jar_dir\com.mq.micro.client.mqttv3.jar
       com.ibm.mq.id.PubSync.java com.ibm.mq.id.Example.java
```

Komutu kullanarak PubSync komutunu çalıştırın.

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.PubSync
```

Yordam

1. Eclipseiçinde bir Java projesi yaratmanızı sağlar.
   a) **Dosya > Yeni > Java projesi** ve bir proje adı yazın. **İleri**'yi tıklatın.

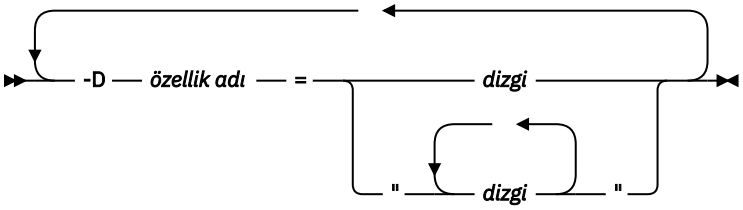
      JRE ' nin doğru ya da sonraki sürümlerde olup olmadığını denetleyin. Java SE 6.0 ya da sonraki bir yayın düzeyinde olmalıdır.
   b) Java Ayarları sayfasında **Kitaplıklar > Dış Jar Ekle ...**öğelerini tıklatın.
   c) WebSphere MQ Telemetry SDK klasörünü yüklediğiniz dizine göz atın. SDK`\clients\java` klasörünü bulun ve tüm `.jar` dosyalarını > **> Aç > Son**seçeneklerini belirleyin.
2. MQTT istemcisi Javadoc 'ı kurun.

   MQTT istemcisi Javadoc kurulu olduğu için, Java düzenleyicisi MQTT v3 sınıflarına ilişkin yardım sağlar.
   a) Java projenizde, **Paket Gezgini > Gönderme Yapılan Kitaplıklar**' ı açın. `com.ibm.micro.client.mqttv3.jar` **> Özellikler**seçeneğini sağ tıklatın.
   b) Özellikler gezgininde **Javadoc Yeri**seçeneğini tıklatın.
   c) Javadoc Yeri sayfasında **Javadoc URL > Göz At ...**öğelerini tıklatın. ve *WMQ Installation directory*`\mqxr\SDK\clients\java\doc\javadoc` klasörünü bulun **> Tamam**.
   d) **Doğrula ... > Tamam**düğmesini tıklatın.

      Belgeleri görüntülemek için bir tarayıcı açmanız istenir.
3. Create the class, PubSync, using the Java Class wizard.
   a) Yarattığınız Java projesini farenin sağ düğmesiyle tıklatın > **Yeni** > **Sınıf**.
   b) Paket adını yazın, `com.ibm.mq.id`
   c) Sınıf adını yazın, PubSync
   d) Yöntem kod parçası kutusunu işaretleyin, **public static void main (String [] args)**

4. `com.ibm.mq.id`paketindeki `Example.java` dosyasını oluşturun. Kodu Şekil 89 sayfa 461 içinde dosyaya kopyalayın.

   Örneklerde kullanılan tüm değiştirgeler özellik olarak ayarlanır. You can override the values by changing the defaults in `Example.java`, or by supplying the properties as options on the Java command line using the `-D` parameter:



   Bu örnekte kullanılan istemci tanıtıcısı ve “Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması” sayfa 461 örnekleri, rasgele bir dizgiyle kullanıcı adı düzeltilir.
5. Kodu oluşturmak için adımları izleyin ya da kodu Şekil 88 sayfa 460olanağından kopyalayın.

   Aşağıdaki adımlarda, `Pubsync.java`içindeki kodu açıkla.
6. Bir `try-catch` bloğu oluşturun.

```
try { ...
 } catch (Exception e) {
  e.printStackTrace();
}
```

   MQTT istemcisi `MqttException`, `MqttPersistenceException` ya da `MqttSecurityException`' i yayınlatabiliyor. `MqttPersistenceException` ve `MqttSecurityException` , `MqttException`alt sınıflarıdır.

   Kural dışı duruma ilişkin nedeni öğrenmek için `MqttException.getReasonCode` yöntemini kullanın. Bir `MqttPersistenceException` ya da `MqttSecurityException` atılırsa, temeldeki atılabilir kural dışı durumu döndürmek için `getCause` yöntemini kullanın.
7. Yeni bir `MqttClient` yönetim ortamı yaratın.

```
MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
```

   İstemciyi daha sonra WebSphere MQ' ya bağlanmak için kullanılan bir sunucu adresi sağlayın. İstemcinin adını vermek için istemci tanıtıcısını ayarlayın.

   - İsteğe bağlı olarak, varsayılan somutlamayı değiştirmek için `MqttClientPersistence` arabiriminin bir somutlamasını sağlayabilirsiniz. Varsayılan `MqttPersistence` uygulaması, dosyalar olarak teslim edilmeyi bekleyen QoS 1 ve 2 iletileri depolar; bkz. “Message persistence in MQTT clients” sayfa 512.
   - MQTT için varsayılan IBM WebSphere MQ TCP/IP kapısı 1883 'tür. SSL için, 8883 'tür. Örnekte, varsayılan adres `tcp://localhost:1883`olarak ayarlıdır.
   - Tipik olarak, istemci tanıtıcısını kullanarak belirli bir fiziksel istemciyi tanımlayabilmek önemlidir. İstemci tanıtıcısı, sunucuya bağlanan tüm istemcilerde benzersiz olmalıdır; bkz. “İstemci tanıtıcısı” sayfa 508. Önceki eşgörünümle aynı istemci tanıtıcısı kullanılması, yürürlükteki yönetim ortamının aynı istemcinin bir yönetim ortamı olduğunu gösterir. Çalışan iki istemcide bir istemci tanıtıcısını kopyaladığınızda, her iki istemciye de bir özel durum oluşur ve bir istemci sonlandırılır.
   - İstemci tanıtıcısının uzunluğu 23 byte ile sınırlanmıştır. Uzunluk aşılırsa, kural dışı durum oluşur. İstemci tanıtıcısı yalnızca, kuyruk yöneticisi adında izin verilen karakterleri içermelidir; örneğin, tire ya da boşluk içermemelidir.
   - `MqttClient.connect` yöntemini çağırıncaya kadar, hiçbir ileti işleme gerçekleşmez.

   Konuları yayınlamak ve abone olmak ve henüz teslim edilmemiş yayınlarla ilgili bilgileri kurtarmak için istemci nesnesini kullanın.

8. Yayınlanacak bir konu oluşturun.

```
MqttTopic topic = client.getTopic(Example.topicString);
```

Bir konu dizgisi, bir IBM WebSphere MQ konu dizgisinin uzunluk üst sınırını aşan 64 K byte ile sınırlıdır. Tersi durumda, bir konu dizgisi, WebSphere MQ konu dizgileriyle aynı kuralları izler; bkz. Konu dizgileri. Örnek, `MQTT Examples`konu dizesini ayarlar.

9. Bir yayın iletisi oluşturun.

```
MqttMessage message = new MqttMessage(Example.publication.getBytes());
```

`"Hello World"` dizgisi byte dizisine dönüştürüldü ve bir `MqttMessage`yaratmak için kullanıldı.

- MQTT ileti bilgi yükü her zaman bayt dizisidir. `getBytes` yöntemi, bir dizgi nesnesini UTF-8biçimine dönüştürür. `MqttMessage` , ileti bilgi yükünü dizgi olarak döndürmek için uygun bir `toString` yönteme sahiptir. Bu, `new string(message.getPayload)`ile eşdeğerdir.
- A publication message is sent to the queue manager with an RFH2 header, and the message data is sent as a `jms-bytes` message.
- İleti nesnesi, hizmet kalitesi ve alıkonan öznitelikler sahiptir. Hizmet kalitesi (QoS), MQTT istemcisi ile kuyruk yöneticisi arasında iletinin ne kadar güvenilir bir şekilde aktarılacağını belirler; bkz. “MQTT istemcisi tarafından sağlanan hizmet nitelikleri” sayfa 517. Bir yayının, gelecekteki aboneler için kuyruk yöneticisi tarafından saklanırsa, alıkonan öznitelik denetimleri. Bir yayın saklanmazsa, yalnızca geçerli abonelere gönderilir; bkz. “Alıkonan yayınlar ve MQTT istemcileri” sayfa 518. Varsayılan `MqttMessage` ayarları şunlardır: "İletiler en az bir kez teslim edilir ve saklanmaz."

10. Sunucuya bağlanın.

```
client.connect();
```

Örnek, varsayılan bağlantı seçeneklerini kullanarak sunucuya bağlanır. Bağlandıktan sonra yayınlamaya başlayabilirsiniz. Varsayılan bağlantı seçenekleri şunlardır:

- TCP/IP bağlantısının kapatılmasını önlemek için, her 15 saniyede bir küçük bir "keep-canlive" (canlı tutma) iletisi gönderilir.
- Oturum, önceki yayınların tamamlanması için denetlenmeden başlatılır.
- Bir iletiyi yeniden göndermeyi denemek arasındaki aralık 15 saniyedir.
- Bağlantı için son bir irade ve ahit iletisi yaratılmadı.
- Bağlantıyı yaratmak için standart SocketFactory kullanılır.

Bir `ConnectionOptions` nesnesi yaratarak ve bunu `client.connect`' a ek bir parametre olarak geçirerek bağlantı seçeneklerini değiştirin.

11. Yayınla.

```
MqttDeliveryToken token = topic.publish(message);
```

Örnek, kuyruk yöneticisine "MQTT Örnekleri" başlıklı konuda "Merhaba Dünya" yayınını gönderir.

- `publish` yöntemi geri döndüğünde, ileti MQTT istemcisine güvenle aktarılır, ancak sunucuya aktarılmaz. İletide QoS 1 ya da 2 varsa, teslim işlemi tamamlanmadan istemcinin başarısız olması durumunda ileti yerel olarak saklanır.
- `publish` , henüz sunucudan alındı bildirimi alınmadığını denetlemek için kullanılan bir teslim belirteci döndürür.

12. Sunucudan onay beklenmesini bekleyin.

```
token.waitForCompletion(Example.timeout);
```

PubSync örneği, iletinin teslim edildiğini onaylayan sunucudan bir alındı bildirimi bekler.

- Zamanaşımı olmadan, istemci sınırsız süreyle bekler. “Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması” sayfa 461 , bir geri bildirme nesnesi kullanarak beklemelerin nasıl alınacağını göstermektedir.

13. İstemciyi sunucudan çıkarın.

```
client.disconnect();
```

    İstemci sunucudan bağlanıyor ve sona ermek için çalışmakta olan tüm `MqttCallback` yöntemlerini bekler. Daha sonra kalan işi bitirmek için 30 saniyeye kadar bekler. Susturma zaman aşımı değerini ek bir değiştirge olarak belirtebilirsiniz.

14. Değişiklikleri `PubSync.java` ve `Example.Java`olarak kaydetme

    Eclipse , Java 'yı otomatik olarak derler. Şimdi programı çalıştırarak sonuçları görmeye hazırsınız.

Sonuçlar

WebSphere MQolanağını kullanarak yayınları görmek için, Şekil 87 sayfa 459dosyasında komut dosyasını kullanarak `"MQTTExampleTopic"` adlı bir konu, bir kuyruk ve kalıcı bir abonelik yaratın. Run the client to publish on the `MQTT Examples` topic, and then run the sample program **amqsbcg** to browse the publications on the `MQTTExamples` queue.

1. Bir kuyruk yöneticisi başlatın ve çalışmakta olan telemetri (MQXR) hizmetini başlatın. Telemetri kanalı için yapılandırılan TCP/IP adresinin ve kapının, MQTT uygulamasında kullandığınız değerlerle eşleştiğinden emin olun.
2. Configure a durable subscription by creating the `mqttexamples.txt` command script, and running it using **runmqsc:**

```
DEFINE TOPIC('MQTTExampleTopic') TOPICSTR('MQTT Example') REPLACE
DEFINE QLOCAL('MQTTExampleQueue') REPLACE
DEFINE SUB('MQTTExampleSub') DEST('MQTTExampleQueue') TOPICOBJ('MQTTExampleTopic') REPLACE
```

Şekil 87. `mqttExampleTopic.txt`

    Komut dosyasını Windows' ta çalıştırmak için şu komutu yazın:

```
runmqsc queue manager name < mqttExampleTopic.txt
```

3. İstemciyi Eclipseiçinden bir Java uygulaması olarak ya da bir komut penceresinde Java çalıştırarak çalıştırın:

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.classname.class
```

    Not: The command window must be open in the directory containing the path, `com\ibm\mq\id`.

4. Either browse the results using WebSphere MQ Explorer, or run the command:

```
amqsbcg MQTTExampleQueue queue manager name
```

Örnek kod

PubSync.java , Yordam' da açıklanan kodun tam listesidir. Modify the `Example` class in Şekil 89 sayfa 461 to override the default parameters used in PubSync.java.

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class PubSync {
  public static void main(String[] args) {
    try {
      MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
      MqttTopic topic = client.getTopic(Example.topicString);
      MqttMessage message = new MqttMessage(Example.publication.getBytes());
      message.setQos(Example.QoS);
      client.connect();
      System.out.println("Waiting for up to " + Example.sleepTimeout / 1000
          + " seconds for publication of \"" + message.toString()
          + "\" with QoS = " + message.getQos());
      System.out.println("On topic \"" + topic.getName()
          + "\" for client instance: \"" + client.getClientId()
          + "\" on address " + client.getServerURI() + "\"");
      MqttDeliveryToken token = topic.publish(message);
      token.waitForCompletion(Example.sleepTimeout);
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" has been received: " + token.isComplete());
      client.disconnect();
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 88. PubSync.java

```
package com.ibm.mq.id;
import java.util.Properties;
import java.util.UUID;
public final class Example {
  public static final String          TCPAddress =
    System.getProperty("TCPAddress", "tcp://localhost:1883");
  public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");
  public static final String            username =
    System.getProperty("username", System.getProperty("user.name"));
  public static final char []           password =
    System.getProperty("password", "Password").toCharArray();
  public static String                  clientId =
  String.format("%-23.23s", username  + "_" +
    System.getProperty("clientId",
    (UUID.randomUUID().toString())).trim()).replace('-', '_');
  public static final String         topicString =
    System.getProperty("topicString", "MQTT Example");
  public static final String         publication =
    System.getProperty("publication", "Hello World " +
  String.format("%tc", System.currentTimeMillis()));
  public static final int         quiesceTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final int           sleepTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final boolean       cleanSession =
    Boolean.parseBoolean(System.getProperty("cleanSession", "false"));
  public static final int                   QoS =
    Integer.parseInt(System.getProperty("QoS", "1"));
  public static final boolean          retained =
    Boolean.parseBoolean(System.getProperty("retained", "false"));
  public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore",
        "C:\\IBM\\MQ\\Data\\ClientKeyStore.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword",
        "password");
    properties.setProperty("com.ibm.ssl.trustStore",
        "C:\\IBM\\MQ\\Data\\ClientTrustStore.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword",
        "password");
    return properties;
  }
}
```

Şekil 89. Example.java

İlgili kavramlar

MQTT yayınlama/abone olma uygulamaları

Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması

Bu görevde, ilk yayınlayıcı uygulamanızı değiştirmek için bir öğretici program izlemeniz gerekir. Değişiklikler, uygulamanın teslim onaylarını beklemeden yayınları göndermesine olanak sağlar. Teslim onayları, oluşturduğunuz bir geri bildirme sınıfı tarafından alınır.

Başlamadan önce

WebSphere MQ Telemetry özelliğini, IBM WebSphere MQ Version 7.1 ya da daha sonraki bir sürümü kurulu olan bir sunucuya kurun.

İstemci uygulaması, IBM WebSphere MQ Telemetry Software Development Toolkit (SDK) içinde `com.ibm.mq.micro.client.mqttv3` paketini kullanır. SDK, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır. The client connects to the IBM WebSphere MQ Telemetry feature to exchange messages with IBM WebSphere MQ.

IBM WebSphere MQ Telemetry' i yönetmek için IBM WebSphere MQ Explorer Version 7.1 ile ilgili telemetri güncellemelerini de kurmalısınız. Güncellemeler, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır.

Java SE üzerinde çalışan bir MQTT istemcisi, Java SE ' nin 6.0 sürümünü ya da üstünü gerektirir. IBM Java SE v6.0 , IBM WebSphere MQ Version 7.1 kuruluşunun bir parçasıdır. Şu konumda bulunur: *WebSphere MQ installation directory*\java\jre

Bu görev hakkında

Örnek, bir yayınlama uygulamasıdır, PubAsync. PubAsync publishes `Hello World` on the topic `MQTT Examples`, without waiting for confirmation that the publication has been delivered to the queue manager. The delivery acknowledgments are received in a callback class, `CallBack`.

By setting up a durable subscription to `MQTT Examples` you can check that the application works.

Yordam, istemciyi geliştirmek, oluşturmak ve çalıştırmak için Eclipse ' i kullanır. You can download Eclipse from the Eclipse project website at www.eclipse.org.

Yordam ' daki adımlar, "Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması" sayfa 455uygulamasında PubSync`.java` uygulamasını değiştirir.

Diğer bir seçenek olarak, kodu "Örnek kod" sayfa 464yeni bir dizine (`.\com\ibm\mq\id`) kopyalayabilirsiniz. Üç Java dosyası oluşturun: `Example.java`, `CallBack.java`ve PubAsync`.java`. Komutu kullanarak örnekleri derleyin,

```
javac -cp jar_dir\com.mq.micro.client.mqttv3.jar
       com.ibm.mq.id.PubAsync.java com.ibm.mq.id.CallBack.java com.ibm.mq.id.Example.java
```

Komutu kullanarak PubAsync komutunu çalıştırın.

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.PubAsync.class
```

Yordam

1. `com.ibm.mq.id` paketinde bir dosya (`CallBack.java`) oluşturun. Kodu Şekil 92 sayfa 464 içinde dosyaya kopyalayın.

   `Callback.java` , `MqttCallBack` arabirimini gerçekleştirir. Örnekte, ek bir oluşturucu geri çağrıyı bazı eşgörünüm verileriyle kullanıma hazırlar.
2. `com.ibm.mq.id` paketinde, PubSync`.java` öğesini farenin sağ düğmesiyle tıklatın ve kopyalayın. Aynı paketi, PubAsyncolarak yeniden adlandırmak için aynı pakete yapıştırın.
3. Just before the `client.connect();` line of code, instantiate the `CallBack` class, passing the client identifier.

   ```
   CallBack callback = new CallBack(Example.clientId);
   client.setCallback(callback);
   ```

   - `CallBack` sınıfı `MqttCallBack`öğesini uygular. İstemci tanıtıcısı başına bir geri çağırma eşgörünümü gerekli. Bu örnekte, oluşturucu eşgörünüm verileri olarak saklanacak istemci tanıtıcısını aktarır. Geri bildirmede, geri çağırma eşgörünümünün hangi yönetim ortamının başlatıldığını saptamak için kullanılır.
   - Geri çağırma sınıfında üç yöntem uygulamanız gerekir:

     `public void messageArrived(MqttTopic topic, MqttMessage message)`
     Abone olunan bir yayını alır.

     `public void connectionLost(Throwable cause)`
     Bağlantı kaybolduğunda çağrılır.

     `public void deliveryComplete(MqttDeliveryToken token))`
     Yayınlanmış bir QoS 1 ya da 2 iletisi için bir teslim simgesi alındığında çağrılır.

- Geri çağırma `MqttClient.connect`tarafından etkinleştirilir.

4. İstemcinin bağlantısını kes
   a) `token.waitForCompletion` ifadesini içeren deyimi kaldırın.

      Ana iş parçacığı, yayınının teslim edilmesini beklemeden devam eder.

   b) İstemcinin önceden bağlantısının kesilip kesilmediğini test edin.

      The MQTT client disconnects following an error returned to the `lostConnection` method in `MqttCallback`, or the client application might disconnect. Açık bir bağlantı olup olmadığını görmek için test edin.

   c) İstemciyi susturma süresi üst sınırını ayarlamak için `Example.quiesceTimeout`değişmezini kullanın.

```
if (client.isConnected())
    client.disconnect(Example.quiesceTimeout);
```

   Müşteri, aşağıdaki üç koşulun bir birleşiminin doğru olduğunu zaman zaman bitirir:

   a. Geri çağırma, bu oturumda yayınlanmış olan tüm iletiler için ya da oturum yeniden başlatıldıysa, önceki oturumlarda çağrılmıştır.
   b. İletiler uçuşta ve susturma aralığının süresi doldu. Varsayılan olarak susturma aralığı 30 saniyedir. You can change the quiesce timeout by passing the number of milliseconds to wait as a parameter of `client.disconnect`.
   c. `client.disconnect` bazı iletiler yayınlandıktan ve istemci tarafından kuyruğa alındıktan sonra, ancak iletiler gönderilmeden önce çağrıldı. Kuyruğa alınan iletiler henüz uçuş sırasında değil. Oturum yeniden başlatılır ise, oturum yeniden başlatıldığında iletiler yeniden gönderilmektedir.

Sonuçlar

WebSphere MQolanağını kullanarak yayınları görmek için, Şekil 90 sayfa 463dosyasında komut dosyasını kullanarak "`MQTTExampleTopic`" adlı bir konu, bir kuyruk ve kalıcı bir abonelik yaratın. Run the client to publish on the `MQTT Examples` topic, and then run the sample program **amqsbcg** to browse the publications on the `MQTTExamples` queue.

1. Bir kuyruk yöneticisi başlatın ve çalışmakta olan telemetri (MQXR) hizmetini başlatın. Telemetri kanalı için yapılandırılan TCP/IP adresinin ve kapının, MQTT uygulamasında kullandığınız değerlerle eşleştiğinden emin olun.
2. Configure a durable subscription by creating the `mqttexamples.txt` command script, and running it using **runmqsc:**

```
DEFINE TOPIC('MQTTExampleTopic') TOPICSTR('MQTT Example') REPLACE
DEFINE QLOCAL('MQTTExampleQueue') REPLACE
DEFINE SUB('MQTTExampleSub') DEST('MQTTExampleQueue') TOPICOBJ('MQTTExampleTopic') REPLACE
```

Şekil 90. mqttExampleTopic.txt

   Komut dosyasını Windows' ta çalıştırmak için şu komutu yazın:

```
runmqsc queue manager name < mqttExampleTopic.txt
```

3. İstemciyi Eclipseiçinden bir Java uygulaması olarak ya da bir komut penceresinde Java çalıştırarak çalıştırın:

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.classname.class
```

   Not: The command window must be open in the directory containing the path, `com\ibm\mq\id`.

4. Either browse the results using WebSphere MQ Explorer, or run the command:

```
amqsbcg MQTTExampleQueue queue manager name
```

Örnek kod

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class PubAsync {
  public static void main(String[] args) {
    try {
      MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
      MqttTopic topic = client.getTopic(Example.topicString);
      MqttMessage message = new MqttMessage(Example.publication.getBytes());
      CallBack callback = new CallBack(Example.clientId);
      client.setCallback(callback);
      client.connect();
      System.out.println("Publishing \"" + message.toString()
          + "\" on topic \"" + topic.getName() + "\" with QoS = "
          + message.getQos());
      System.out.println("For client instance \"" + client.getClientId()
          + "\" on address " + client.getServerURI() + "\"");
      MqttDeliveryToken token = topic.publish(message);
      System.out.println("With delivery token \"" + token.hashCode()
          + " delivered: " + token.isComplete());
      if (client.isConnected())
          client.disconnect(Example.quiesceTimeout);
      System.out.println("Disconnected: delivery token \"" + token.hashCode()
          + "\" received: " + token.isComplete());
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 91. PubAsync.java

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class CallBack implements MqttCallback {
  private String instanceData = "";
  public CallBack(String instance) {
    instanceData = instance;
  }
  public void messageArrived(MqttTopic topic, MqttMessage message) {
    try {
      System.out.println("Message arrived: \"" + message.toString()
          + "\" on topic \"" + topic.toString() + "\" for instance \""
          + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
  public void connectionLost(Throwable cause) {
    System.out.println("Connection lost on instance \"" + instanceData
        + "\" with cause \"" + cause.getMessage() + "\" Reason code "
        + ((MqttException)cause).getReasonCode() + "\" Cause \""
        + ((MqttException)cause).getCause() +  "\"");
    cause.printStackTrace();
  }
  public void deliveryComplete(MqttDeliveryToken token) {
    try {
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" received by instance \"" + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 92. CallBack.java

```
package com.ibm.mq.id;
import java.util.Properties;
import java.util.UUID;
public final class Example {
  public static final String          TCPAddress =
    System.getProperty("TCPAddress", "tcp://localhost:1883");
  public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");
  public static final String            username =
    System.getProperty("username", System.getProperty("user.name"));
  public static final char []           password =
    System.getProperty("password", "Password").toCharArray();
  public static String                  clientId =
  String.format("%-23.23s", username  + "_" +
    System.getProperty("clientId",
    (UUID.randomUUID().toString())).trim()).replace('-', '_');
  public static final String         topicString =
    System.getProperty("topicString", "MQTT Example");
  public static final String         publication =
    System.getProperty("publication", "Hello World " +
  String.format("%tc", System.currentTimeMillis()));
  public static final int         quiesceTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final int           sleepTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final boolean       cleanSession =
    Boolean.parseBoolean(System.getProperty("cleanSession", "false"));
  public static final int                   QoS =
    Integer.parseInt(System.getProperty("QoS", "1"));
  public static final boolean           retained =
    Boolean.parseBoolean(System.getProperty("retained", "false"));
  public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore",
        "C:\\IBM\\MQ\\Data\\ClientKeyStore.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword",
        "password");
    properties.setProperty("com.ibm.ssl.trustStore",
        "C:\\IBM\\MQ\\Data\\ClientTrustStore.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword",
        "password");
    return properties;
  }
}
```

Şekil 93. Example.java

Java kullanan MQ Telemetry Transport için kurtarılabilir bir zamanuyumsuz yayınlayıcı yaratılması

Bu görevde, zamanuyumsuz yayınlayıcı uygulamanızı değiştirmek için bir öğretici program izlemeniz gerekir. Değişiklikler, uygulamanın, istemcinin en son çalıştırıldığı anda onaylanmamış yayınların teslim işlemini tamamlamaya olanak sağlar.

Başlamadan önce

WebSphere MQ Telemetry özelliğini, IBM WebSphere MQ Version 7.1 ya da daha sonraki bir sürümü kurulu olan bir sunucuya kurun.

İstemci uygulaması, IBM WebSphere MQ Telemetry Software Development Toolkit (SDK) içinde `com.ibm.mq.micro.client.mqttv3` paketini kullanır. SDK, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır. The client connects to the IBM WebSphere MQ Telemetry feature to exchange messages with IBM WebSphere MQ.

IBM WebSphere MQ Telemetry' i yönetmek için IBM WebSphere MQ Explorer Version 7.1 ile ilgili telemetri güncellemelerini de kurmalısınız. Güncellemeler, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır.

Java SE üzerinde çalışan bir MQTT istemcisi, Java SE ' nin 6.0 sürümünü ya da üstünü gerektirir. IBM Java SE v6.0 , IBM WebSphere MQ Version 7.1 kuruluşunun bir parçasıdır. Şu konumda bulunur: *WebSphere MQ installation directory*\java\jre

Bu görev hakkında

Örnek, bir yayınlama uygulamasıdır, PubAsyncRestartable. PubAsyncRestartable publishes Hello World on the topic MQTT Examples, without waiting for confirmation that the publication has been delivered to the queue manager. The delivery acknowledgments are received in a callback class, CallBack. Önceki bir örnekte tamamlanmamış yayınlar için teslim simgeleri incelenebilir. Bunlar, geri çağırma sınıfı tarafından da işlenir.

By setting up a durable subscription to MQTT Examples you can check that the application works.

Yordam, istemciyi geliştirmek, oluşturmak ve çalıştırmak için Eclipse ' i kullanır. You can download Eclipse from the Eclipse project website at www.eclipse.org.

Yordam ' daki adımlar, “Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması” sayfa 461uygulamasında PubAsync.java uygulamasını değiştirir.

Diğer bir seçenek olarak, kodu “Örnek kod” sayfa 469yeni bir dizine (.\com\ibm\mq\id) kopyalayabilirsiniz. Üç Java dosyası oluşturun: Example.java, CallBack.javave PubAsyncRestartable.java. Komutu kullanarak örnekleri derleyin,

```
javac -cp jar_dir\com.mq.micro.client.mqttv3.jar
       com.ibm.mq.id.PubAsyncRestartable.java com.ibm.mq.id.CallBack.java
com.ibm.mq.id.Example.java
```

Komutu kullanarak PubAsyncRestartable komutunu çalıştırın.

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.PubAsyncRestartable.class
```

Yordam

1. com.ibm.mq.id paketinde, PubAsync.java öğesini farenin sağ düğmesiyle tıklatın ve kopyalayın. Aynı paketi, PubAsyncRestartableolarak yeniden adlandırmak için aynı pakete yapıştırın.
2. Yeniden kullanılabilir bir istemci tanıtıcısı yaratın.

   ```
   Example.clientId = String.format(
       "%-23.23s",
       (System.getProperty("user.name") + "_" + (System.getProperty(
           "clientId", "PubAsyncRestartable."))).trim()).replace('-', '_');
   ```

   Şekil 94. Yeniden kullanılabilir istemci tanıtıcısı

   “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 ve “Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması” sayfa 461 içindeki uygulamalar, her istemci bağlantısı için yeni bir istemci tanıtıcısı kullandı. Yeniden başlatılabilir bir yayınlayıcı ya da abone için, istemci her bağlandığınızda aynı istemci tanıtıcısını kullanmanız gerekir, ancak farklı istemciler farklı tanıtıcılar kullanmalıdır; bkz. “İstemci tanıtıcısı” sayfa 508. Yeniden kullanılabilir istemci tanıtıcısı, kullanıcı adından ve sınıfın adından oluşturulur. Bu, 23 bayt ile sınırlıdır. Yalnızca kuyruk yöneticisi nesne adlarında geçerli karakterler olmalıdır. Kod, yerleştirmiş olabilecek tire işaretlerini kaldırır.
3. Yinelenen iletileri önlemek için, iletinin QoS değeri, varsayılan değer olan 1 yerine 2 değerine ayarlanır.

   ```
   message.setQos(Example.QoS);
   ```

Java komut satırındaki -DQoS=2 seçeneğini kullanarak Example.QoS değerini 2olarak değiştirmeniz ya da QoS özelliğini bağımsız değişken olarak geçirmeniz gerekir.

4. Create an MqttConnectOptions object, and set its cleanSession attribute to false.

   a) Bir MqttConnectOptions nesnesi oluşturun.

   ```
   MqttConnectOptions conOptions = new MqttConnectOptions();
   ```

   conOptions , MqttClient oluşturucuda bir seçenek parametresidir.

   b) clearSession özniteliğini ayarlayın.

   ```
   conOptions.setCleanSession(Example.cleanSession);
   ```

   By default, the parameter Example.cleanSession is set to true, matching the default setting of MqttConnectionOptions.cleanSession.

   PubAsyncRestartable yeniden başlatıldığında, "temiz bir oturumla" başlayabilir ve QoS 1 ya da 2 'ye ilişkin iletiler için bekleyen teslim simgeleri temizleyebilir.

   Set Example.cleanSession to false to keep all the pending delivery tokens. Simgeler, istemci yeniden bağlandığında MqttCallBack sınıfı tarafından işlenir.

5. Oturum yeniden başlatılıyorsa, beklemedeki teslim simgelerini alın ve içindekileri yazdırın.

```
if (!conOptions.isCleanSession()) {
        MqttDeliveryToken tokens[] = client.getPendingDeliveryTokens();
        System.out.println("Starting a previous session for instance \""
            + client.getClientId() + "\" with " + tokens.length
            + " delivery tokens pending");
        for (int i = 0; i < tokens.length; i++) {
          System.out.println("Message \"" + tokens[i].getMessage().toString()
              + "\" with QoS=" + tokens[i].getMessage().getQos()
              + " recovered by instance \"" + client.getClientId()
              + "\" and assigned delivery token \"" + tokens[i].hashCode()
              + "\"");
        }
      } else
        System.out.println("Starting a clean session for instance "
            + client.getClientId());
```

6. conOptions parametresini MqttClient oluşturucusuna geçirin.

```
client.connect(conOptions);
```

7. Bağlantı kesiliyor, bir bağlantı kesme aralığı üst sınırı belirleyin.

```
client.disconnect(Example.timeout);
```

İşlenmekte olan bekleyen teslim belirteçlerini gösterebilmek için, önceki bir eşgörünüm teslimi tamamlamadan bitmelidir. Örneğin, yayınları PubAsyncRestartable sona ermeden önce kabul etmemek olasılığına sahip olarak çalıştırmak için, Example.timeout seçeneğini 0 olarak ayarlayın.

Sonuçlar

WebSphere MQolanağını kullanarak yayınları görmek için, Şekil 95 sayfa 468dosyasında komut dosyasını kullanarak "MQTTExampleTopic" adlı bir konu, bir kuyruk ve kalıcı bir abonelik yaratın. Run the client to publish on the MQTT Examples topic, and then run the sample program **amqsbcg** to browse the publications on the MQTTExamples queue.

1. Bir kuyruk yöneticisi başlatın ve çalışmakta olan telemetri (MQXR) hizmetini başlatın. Telemetri kanalı için yapılandırılan TCP/IP adresinin ve kapının, MQTT uygulamasında kullandığınız değerlerle eşleştiğinden emin olun.
2. Configure a durable subscription by creating the mqttexamples.txt command script, and running it using **runmqsc:**

```
DEFINE TOPIC('MQTTExampleTopic') TOPICSTR('MQTT Example') REPLACE
DEFINE QLOCAL('MQTTExampleQueue') REPLACE
DEFINE SUB('MQTTExampleSub') DEST('MQTTExampleQueue') TOPICOBJ('MQTTExampleTopic') REPLACE
```

Şekil 95. mqttExampleTopic.txt

Komut dosyasını Windows' ta çalıştırmak için şu komutu yazın:

```
runmqsc queue manager name < mqttExampleTopic.txt
```

3. İstemciyi Eclipseiçinden bir Java uygulaması olarak ya da bir komut penceresinde Java çalıştırarak çalıştırın:

```
java -cp jar_dir\com.mq.micro.client.mqttv3.jar
      com.ibm.mq.id.classname.class
```

Not: The command window must be open in the directory containing the path, `com\ibm\mq\id`.

4. Either browse the results using WebSphere MQ Explorer, or run the command:

```
amqsbcg MQTTExampleQueue queue manager name
```

Örnek kod

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.MqttClient;
import com.ibm.micro.client.mqttv3.MqttConnectOptions;
import com.ibm.micro.client.mqttv3.MqttDeliveryToken;
import com.ibm.micro.client.mqttv3.MqttMessage;
import com.ibm.micro.client.mqttv3.MqttTopic;
public class PubAsyncRestartable {
  public static void main(String[] args) {
    Example.clientId = String.format(
        "%-23.23s",
        (System.getProperty("user.name") + "_" + (System.getProperty(
            "clientId", "PubAsyncRestartable."))).trim()).replace('-', '_');
    try {
      MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
      MqttTopic topic = client.getTopic(Example.topicString);
      MqttMessage message = new MqttMessage(Example.publication.getBytes());
      message.setQos(Example.QoS);
      CallBack callback = new CallBack(Example.clientId);
      client.setCallback(callback);
      MqttConnectOptions conOptions = new MqttConnectOptions();
      conOptions.setCleanSession(Example.cleanSession);
      if (!conOptions.isCleanSession()) {
        MqttDeliveryToken tokens[] = client.getPendingDeliveryTokens();
        System.out.println("Starting a previous session for instance \""
            + client.getClientId() + "\" with " + tokens.length
            + " delivery tokens pending");
        for (int i = 0; i < tokens.length; i++) {
          System.out.println("Message \"" + tokens[i].getMessage().toString()
              + "\" with QoS=" + tokens[i].getMessage().getQos()
              + " recovered by instance \"" + client.getClientId()
              + "\" and assigned delivery token \"" + tokens[i].hashCode()
              + "\"");
        }
      } else
        System.out.println("Starting a clean session for instance \""
            + client.getClientId() + "\"");
      client.connect(conOptions);
      System.out.println("Publishing \"" + message.toString()
          + "\" on topic \"" + topic.getName() + "\" with QoS = "
          + message.getQos());
      System.out.println("For client instance \"" + client.getClientId()
          + "\" on address " + client.getServerURI() + "\"");
      MqttDeliveryToken token = topic.publish(message);
      System.out.println("With delivery token \"" + token.hashCode()
          + " delivered: " + token.isComplete());
      if (client.isConnected())
          client.disconnect(Example.quiesceTimeout);
      System.out.println("Disconnected: delivery token \"" + token.hashCode()
          + "\" has been received: " + token.isComplete());
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 96. PubAsyncRestartable.java

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class CallBack implements MqttCallback {
  private String instanceData = "";
  public CallBack(String instance) {
    instanceData = instance;
  }
  public void messageArrived(MqttTopic topic, MqttMessage message) {
    try {
      System.out.println("Message arrived: \"" + message.toString()
          + "\" on topic \"" + topic.toString() + "\" for instance \""
          + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
  public void connectionLost(Throwable cause) {
    System.out.println("Connection lost on instance \"" + instanceData
        + "\" with cause \"" + cause.getMessage() + "\" Reason code "
        + ((MqttException)cause).getReasonCode() + "\" Cause \""
        + ((MqttException)cause).getCause() +  "\"");
    cause.printStackTrace();
  }
  public void deliveryComplete(MqttDeliveryToken token) {
    try {
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" received by instance \"" + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 97. CallBack.java

```
package com.ibm.mq.id;
import java.util.Properties;
import java.util.UUID;
public final class Example {
  public static final String          TCPAddress =
    System.getProperty("TCPAddress", "tcp://localhost:1883");
  public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");
  public static final String            username =
    System.getProperty("username", System.getProperty("user.name"));
  public static final char []           password =
    System.getProperty("password", "Password").toCharArray();
  public static String                  clientId =
  String.format("%-23.23s", username  + "_" +
    System.getProperty("clientId",
    (UUID.randomUUID().toString())).trim()).replace('-', '_');
  public static final String         topicString =
    System.getProperty("topicString", "MQTT Example");
  public static final String         publication =
    System.getProperty("publication", "Hello World " +
  String.format("%tc", System.currentTimeMillis()));
  public static final int         quiesceTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final int           sleepTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final boolean       cleanSession =
    Boolean.parseBoolean(System.getProperty("cleanSession", "false"));
  public static final int                  QoS =
    Integer.parseInt(System.getProperty("QoS", "1"));
  public static final boolean           retained =
    Boolean.parseBoolean(System.getProperty("retained", "false"));
  public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore",
        "C:\\IBM\\MQ\\Data\\ClientKeyStore.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword",
        "password");
    properties.setProperty("com.ibm.ssl.trustStore",
        "C:\\IBM\\MQ\\Data\\ClientTrustStore.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword",
        "password");
    return properties;
  }
}
```

Şekil 98. Example.java

Java kullanılarak MQ Telemetry Transport için abone yaratılması

Bu görevde, bir abone uygulaması yaratmak için bir öğretici program izlemenizi sağlar. Abone, bir konuya ilişkin abonelik oluşturur ve aboneliğe ilişkin yayınları alır.

Başlamadan önce

WebSphere MQ Telemetry özelliğini, IBM WebSphere MQ Version 7.1 ya da daha sonraki bir sürümü kurulu olan bir sunucuya kurun.

İstemci uygulaması, IBM WebSphere MQ Telemetry Software Development Toolkit (SDK) içinde `com.ibm.mq.micro.client.mqttv3` paketini kullanır. SDK, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır. The client connects to the IBM WebSphere MQ Telemetry feature to exchange messages with IBM WebSphere MQ.

IBM WebSphere MQ Telemetry' i yönetmek için IBM WebSphere MQ Explorer Version 7.1 ile ilgili telemetri güncellemelerini de kurmalısınız. Güncellemeler, IBM WebSphere MQ Telemetry kuruluşunun bir parçasıdır.

Java SE üzerinde çalışan bir MQTT istemcisi, Java SE ' nin 6.0 sürümünü ya da üstünü gerektirir. IBM Java SE v6.0 , IBM WebSphere MQ Version 7.1 kuruluşunun bir parçasıdır. Şu konumda bulunur: *WebSphere MQ installation directory*\java\jre

Bu görev hakkında

Örnek, bir abone uygulaması (Subscribe). Subscribe creates a subscription topic, MQTT Examples, and waits for publications on the subscription for 30 seconds.

Abone, abonelik yaratabilir ve yayınlar için bekleyebilirsiniz. Ayrıca, aynı istemci tanıtıcısı için daha önce yaratılmış bir aboneliğe gönderilen yayınları da alabilir. MqttConnectionOptions.cleanSession Boole özniteliği daha önce gönderilen yayınların alınıp alınmayacağını denetler; bkz. “Abonelikler” sayfa 519.

Yayınları oluşturmak için yayınlama örneği programlarını kullanabilir ya da MQTT Examples konusunda bir test yayını oluşturmak için WebSphere MQ gezginini kullanabilirsiniz.

Yordam, istemciyi geliştirmek, oluşturmak ve çalıştırmak için Eclipse ' i kullanır. You can download Eclipse from the Eclipse project website at www.eclipse.org.

Yordam ' daki yönergeler, com.ibm.mq.id paketini önceden önceki görevlerden birinde oluşturduğunu varsayar ve Example.java ve Callback.java sınıflarına kopyaladığınızı varsayar.

Yordam

1. Create the class, Subscribe in the package com.ibm.mq.id.
2. Yeniden kullanılabilir bir istemci tanıtıcısı yaratın.

   ```
   Example.clientId = String.format(
       "%-23.23s",
       (System.getProperty("user.name") + "_" + (System.getProperty(
           "clientId", "Subscribe."))).trim()).replace('-', '_');
   ```

   Şekil 99. Yeniden kullanılabilir istemci tanıtıcısı

   “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 ve “Java kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı yaratılması” sayfa 461 içindeki uygulamalar, her istemci bağlantısı için yeni bir istemci tanıtıcısı kullandı. Yeniden başlatılabilir bir yayınlayıcı ya da abone için, istemci her bağlandığınızda aynı istemci tanıtıcısını kullanmanız gerekir, ancak farklı istemciler farklı tanıtıcılar kullanmalıdır; bkz. “İstemci tanıtıcısı” sayfa 508. Yeniden kullanılabilir istemci tanıtıcısı, kullanıcı adından ve sınıfın adından oluşturulur. Bu, 23 bayt ile sınırlıdır. Yalnızca kuyruk yöneticisi nesne adlarında geçerli karakterler olmalıdır. Kod, yerleştirmiş olabilecek tire işaretlerini kaldırır.
3. Bir try-catch bloğu oluşturun.

   ```
   try { ...
    } catch (Exception e) {
     e.printStackTrace();
   }
   ```

   MQTT istemcisi MqttException, MqttPersistenceException ya da MqttSecurityException' i yayınlatabiliyor. MqttPersistenceException ve MqttSecurityException , MqttExceptionalt sınıflarıdır.

   Kural dışı duruma ilişkin nedeni öğrenmek için MqttException.getReasonCode yöntemini kullanın. Bir MqttPersistenceException ya da MqttSecurityException atılırsa, temeldeki atılabilir kural dışı durumu döndürmek için getCause yöntemini kullanın.
4. Yeni bir MqttClient yönetim ortamı yaratın.

   ```
   MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
   ```

   İstemciyi daha sonra WebSphere MQ' ya bağlanmak için kullanılan bir sunucu adresi sağlayın. İstemcinin adını vermek için istemci tanıtıcısını ayarlayın.

- İsteğe bağlı olarak, varsayılan somutlamayı değiştirmek için `MqttClientPersistence` arabiriminin bir somutlamasını sağlayabilirsiniz. Varsayılan `MqttPersistence` uygulaması, dosyalar olarak teslim edilmeyi bekleyen QoS 1 ve 2 iletileri depolar; bkz. “Message persistence in MQTT clients” sayfa 512.
- MQTT için varsayılan IBM WebSphere MQ TCP/IP kapısı 1883 'tür. SSL için, 8883 'tür. Örnekte, varsayılan adres `tcp://localhost:1883`olarak ayarlıdır.
- Tipik olarak, istemci tanıtıcısını kullanarak belirli bir fiziksel istemciyi tanımlayabilmek önemlidir. İstemci tanıtıcısı, sunucuya bağlanan tüm istemcilerde benzersiz olmalıdır; bkz. “İstemci tanıtıcısı” sayfa 508. Önceki eşgörünümle aynı istemci tanıtıcısı kullanılması, yürürlükteki yönetim ortamının aynı istemcinin bir yönetim ortamı olduğunu gösterir. Çalışan iki istemcide bir istemci tanıtıcısını kopyaladığınızda, her iki istemciye de bir özel durum oluşur ve bir istemci sonlandırılır.
- İstemci tanıtıcısının uzunluğu 23 byte ile sınırlanmıştır. Uzunluk aşılırsa, kural dışı durum oluşur. İstemci tanıtıcısı yalnızca, kuyruk yöneticisi adında izin verilen karakterleri içermelidir; örneğin, tire ya da boşluk içermemelidir.
- `MqttClient.connect` yöntemini çağırıncaya kadar, hiçbir ileti işleme gerçekleşmez.

Konuları yayınlamak ve abone olmak ve henüz teslim edilmemiş yayınlarla ilgili bilgileri kurtarmak için istemci nesnesini kullanın.

5. Just before the `client.connect();` line of code, instantiate the `CallBack` class, passing the client identifier.

```
CallBack callback = new CallBack(Example.clientId);
client.setCallback(callback);
```

   - `CallBack` sınıfı `MqttCallBack`öğesini uygular. İstemci tanıtıcısı başına bir geri çağırma eşgörünümü gerekli. Bu örnekte, oluşturucu eşgörünüm verileri olarak saklanacak istemci tanıtıcısını aktarır. Geri bildirmede, geri çağırma eşgörünümünün hangi yönetim ortamının başlatıldığını saptamak için kullanılır.
   - Geri çağırma sınıfında üç yöntem uygulamanız gerekir:

     `public void messageArrived(MqttTopic topic, MqttMessage message)`
     Abone olunan bir yayını alır.

     `public void connectionLost(Throwable cause)`
     Bağlantı kaybolduğunda çağrılır.

     `public void deliveryComplete(MqttDeliveryToken token))`
     Yayınlanmış bir QoS 1 ya da 2 iletisi için bir teslim simgesi alındığında çağrılır.
   - Geri çağırma `MqttClient.connect`tarafından etkinleştirilir.
6. Create an `MqttConnectOptions` object, and set its `cleanSession` attribute.
   a) Bir `MqttConnectOptions` nesnesi oluşturun.

```
MqttConnectOptions conOptions = new MqttConnectOptions();
```

   `conOptions` , `MqttClient` oluşturucuda bir seçenek parametresidir.

   b) clearSession özniteliğini ayarlayın.

```
conOptions.setCleanSession(Example.cleanSession);
```

   By default, the parameter `Example.cleanSession` is set to `true`, matching the default setting of `MqttConnectionOptions.cleanSession`.

Varsayılan `MqttConnectOptions`değerini kullanırsanız ya da istemciyi bağlamadan önce `MqttConnectOptions.cleanSession` değerini `true` olarak ayarlıyorsanız, istemci bağlandığında istemciye ilişkin eski abonelikler kaldırılır. İstemcinin oturum sırasında yaptığı yeni abonelikler bağlantısı kesildiğinde kaldırılır.

If you set `MqttConnectOptions.cleanSession` to `false` before connecting, any subscriptions the client creates are added to all the subscriptions that existed for the client before it connected. İstemci bağlantısı kesildiğinde, tüm abonelikler etkin kalır.

`cleanSession` özniteliğinin abonelikleri etkilemesinin diğer bir yolu da bunu bir kalıcı öznitelik olarak algılamak olur. In its default mode, `cleanSession=true`, the client creates subscriptions and receives publications only within the scope of the session. Alternatif modda `cleanSession=false`, abonelikler dayanıklıdır. İstemci bağlanabilir ve bağlantı kesilebilir ve abonelikleri etkin kalmaya devam eder. İstemci yeniden bağlandığında, teslim edilmemiş yayınlar alır. Bağlantı bağlıyken, kendi adına etkin olan abonelikler kümesini değiştirebilir.

Bağlanmadan önce `cleanSession` kipini ayarlamanız gerekir; kip tüm oturum için geçerli olur. Ayarını değiştirmek için bağlantıyı kesmeniz ve istemciyi yeniden bağlamanız gerekir. If you change modes from using `cleanSession=false` to `cleanSession=true`, all previous subscriptions for the client, and any publications that have not been received, are discarded.

7. `conOptions` parametresini `MqttClient` oluşturucusuna geçirin.

```
client.connect(conOptions);
```

8. Bir abonelik oluşturun.

```
      client.subscribe(Example.topicString, Example.QoS);
```

Bu örnek, bir konu süzgecini `QoS` seçeneğiyle geçen bir `MqttClient.subscribe` yöntemini kullanır. `MqttClient.subscribe` yöntemi dört imzaya sahiptir ve abonelik süzgeçlerinin dizilerini tek bir süzgeç olarak geçirebilirsiniz.

Bu örnek, yayınlama örnekleri tarafından bir konu süzgeci olarak kullanılan konu dizesini kullanır, bu nedenle oluşturdukları her yayını alır.

Örneği her çalıştırdığınızda, `subscribe.java`, bir abonelik oluşturur. Example.topicStringseçeneğini değiştirmezseniz, aynı aboneliği yeniden yaratır. Bir abonelik yeniden yaratılırsa, bu abonelik iki özdeş abonelikle sonuçlanmaz. Bir istemci, aynı abonelikle eşleşen yayınların yinelenen kopyalarını almaz.

Abonelikler “Abonelikler” sayfa 519içinde açıklanır ve “Topic strings and topic filters in MQTT clients” sayfa 521içinde süzgeçler.

9. Bazı yayınların varmasını bekleyin ve daha sonra, istemciyi çıkarın.

```
      Thread.sleep(Example.sleepTimeout);
      client.disconnect();
```

Yayınlar, `MqttCallback.messageArrived` yönteminin somutlaması tarafından alınır.

Abone olma uygulaması herhangi bir ileti yayınlamadı ve bu nedenle teslim simgeleri beklemiyor. `client.disconnect` , herhangi bir gecikme olmadan gerçekleşir.

Örnek kod

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.MqttClient;
import com.ibm.micro.client.mqttv3.MqttConnectOptions;
public class Subscribe {
  public static void main(String[] args) {
    Example.clientId = String.format(
        "%-23.23s",
        (System.getProperty("user.name") + "_" + System.getProperty("clientId",
            "Subscribe."))).trim();
    try {
      MqttClient client = new MqttClient(Example.TCPAddress, Example.clientId);
      CallBack callback = new CallBack(Example.clientId);
      client.setCallback(callback);
      MqttConnectOptions conOptions = new MqttConnectOptions();
      conOptions.setCleanSession(Example.cleanSession);
      client.connect(conOptions);
      System.out.println("Subscribing to topic \"" + Example.topicString
          + "\" for client instance \"" + client.getClientId()
          + "\" using QoS " + Example.QoS + ". Clean session is "
          + Example.cleanSession);
      client.subscribe(Example.topicString, Example.QoS);
      System.out.println("Going to sleep for " + Example.sleepTimeout / 1000
          + " seconds");
      Thread.sleep(Example.sleepTimeout);
      client.disconnect();
      System.out.println("Finished");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 100. Subscribe.java

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class CallBack implements MqttCallback {
  private String instanceData = "";
  public CallBack(String instance) {
    instanceData = instance;
  }
  public void messageArrived(MqttTopic topic, MqttMessage message) {
    try {
      System.out.println("Message arrived: \"" + message.toString()
          + "\" on topic \"" + topic.toString() + "\" for instance \""
          + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
  public void connectionLost(Throwable cause) {
    System.out.println("Connection lost on instance \"" + instanceData
        + "\" with cause \"" + cause.getMessage() + "\" Reason code "
        + ((MqttException)cause).getReasonCode() + "\" Cause \""
        + ((MqttException)cause).getCause() +  "\"");
    cause.printStackTrace();
  }
  public void deliveryComplete(MqttDeliveryToken token) {
    try {
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" received by instance \"" + instanceData + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Şekil 101. CallBack.java

```
package com.ibm.mq.id;
import java.util.Properties;
import java.util.UUID;
public final class Example {
  public static final String          TCPAddress =
    System.getProperty("TCPAddress", "tcp://localhost:1883");
  public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");
  public static final String            username =
    System.getProperty("username", System.getProperty("user.name"));
  public static final char []           password =
    System.getProperty("password", "Password").toCharArray();
  public static String                  clientId =
  String.format("%-23.23s", username  + "_" +
    System.getProperty("clientId",
    (UUID.randomUUID().toString())).trim()).replace('-', '_');
  public static final String         topicString =
    System.getProperty("topicString", "MQTT Example");
  public static final String         publication =
    System.getProperty("publication", "Hello World " +
  String.format("%tc", System.currentTimeMillis()));
  public static final int         quiesceTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final int           sleepTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final boolean       cleanSession =
    Boolean.parseBoolean(System.getProperty("cleanSession", "false"));
  public static final int                   QoS =
    Integer.parseInt(System.getProperty("QoS", "1"));
  public static final boolean           retained =
    Boolean.parseBoolean(System.getProperty("retained", "false"));
  public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore",
        "C:\\IBM\\MQ\\Data\\ClientKeyStore.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword",
        "password");
    properties.setProperty("com.ibm.ssl.trustStore",
        "C:\\IBM\\MQ\\Data\\ClientTrustStore.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword",
        "password");
    return properties;
  }
}
```

Şekil 102. Example.java

İlgili kavramlar

MQTT yayınlama/abone olma uygulamaları

Authenticating an MQTT Java client using JAAS

Learn how to authenticate a client using JAAS. Modify the sample program `JAASLoginModule.java` and the example Java program `PubSync.java`. Configure a telemetry channel to require JAAS authentication, and run the modified publisher, checking its `kullanıcı adı` and `parola` using JAAS.

Başlamadan önce

You are assumed to have installed the MQTT v3 client jar files, Javadoc, Eclipse, configured telemetry channels and coded and run PubSync.java before performing this task. You have an Eclipse workspace that includes a running version of PubSync.java.

Görev Windows için yazılmıştır. Linuxiçin dizin yollarını değiştirin.

Bu görev hakkında

The task is based on modifying the sample JAASLoginModule class in *WMQ Installation directory*\mqxr\samples\JAASLoginModule.java to create MyLogin.java. Ayrıca, bir kullanıcı adı ve parola' yi ayarlamak için "Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması" sayfa 455 içindeki PubSync.java örnek kodunu da değiştirdiniz. As a test, MyLogin.java randomly accepts or rejects kullanıcı adı and parola.

Görevdeki adımlar, programlama alıştırıcı olarak yazılır. Bir üretim ortamında gerçek kimlik doğrulaması gerçekleştirmek için yordamı uyarlamanız gerekir.

JAAS kimlik doğrulamasının nasıl programladığına ilişkin tipik bir açıklamada, oturum açma biriminin JAAS' ı yükleyen bağlamın doğrulamasında olduğu varsayılır. Telemetri (MQXR) hizmeti JAASçağrıldığında, JAAS ' yi yükleyen bağlam telemetri (MQXR) hizmetidir. Telemetri (MQXR) hizmeti bağlamının doğrulamasında bir nokta yoktur; her zaman mqm' dir. Bunun yerine, telemetri (MQXR) hizmeti, oturum açma modülü sınıfına sunulacak istemci kullanıcı adı ve parola istemcisini ayarlar. kullanıcı adı ve parola , iki geri arama kullanarak oturum açma modülüne geçirilir.

```
javax.security.auth.callback.Callback[] callbacks =
      new javax.security.auth.callback.Callback[2];
callbacks[0] =
      new javax.security.auth.callback.NameCallback("NameCallback");
callbacks[1] =
      new javax.security.auth.callback.PasswordCallback("PasswordCallback", false);
callbackHandler.handle(callbacks);
String username =
      ((javax.security.auth.callback.NameCallback) callbacks[0]).getName();
char[] password =
      ((javax.security.auth.callback.PasswordCallback) callbacks[1]).getPassword();
```

İstemcinin kullanıcı adı ve parolası , oturum açma modülü tarafından kullanılabilen istemciye ilişkin tek bilgilerdir.

Yordam

1. Create two packages, samples and security.jaas in the same Java project as PubSync.java.

   samples paketi yalnızca başvuru için kullanılır. Make the code changes in the security.jaas package.
2. JAASLoginModule.java ve JAASPrincipal.java dosyasını her iki pakete de aktarın.

   Gerekiyorsa, derleme hatalarını ortadan kaldırmak için Java kaynağındaki paket deyimlerini yeniden düzenleyin.
3. Refactor the class name, JAASLoginModule, in the security.jaas package to MyLogin
4. MyLogin.java' ta, login yöntemindeki bazı kodları, modülün çalıştığını göstermek içinreplaceyönteminde değiştirin.
   a) Kodu değiştirin:

   ```
   // Accept everything.
   if (true)
     loggedIn = true;
   else
     throw new javax.security.auth.login.FailedLoginException("Login failed");
   ```

   b) Kodla:

   ```
   // login half the users randomly
   PrintWriter pw = new PrintWriter(new FileWriter(System.getProperty("user.dir")
           + "\\MyLogin.log", true));
   pw.println("Called JAASLogin.login at "
           + System.getProperty("publication", "Hello World "
           + String.format("%tc", System.currentTimeMillis())));
   if (Math.random() < 0.5)
     loggedIn = true;
   pw.println("Username: \"" + username + "\", Password: \""
       + String.valueOf(password) + "\" loggedIn: " + loggedIn);
   pw.close();
   if (!loggedIn)
   ```

```
  throw new javax.security.auth.login.FailedLoginException("Login failed");
principal= new JAASPrincipal(username);
```

MyLogin.java için tam kaynak Şekil 105 sayfa 480içinde yer alıyor. The source for JAASPrincipal.java, with the package name refactored to security.jaas is in Şekil 106 sayfa 481.

5. service.env içindeki sınıf yolunu, security/jaas/MyLogin.class ve security/jaas/JAASPrincipal.classyolunu içeren dizine işaret edecek şekilde ayarlayın.

```
CLASSPATH=C:\WMQTelemetryApps\MQTTSecureExamples\bin
```

Bir sınıf yolunu WebSphere MQ hizmetine geçirmek için service.env kullanımıyla ilgili bilgi için Telemetri kanalı JAAS yapılandırması başlıklı konuya bakın.

6. jaas.config' a bir oturum açma modülü kısmı ekleyin.

```
MyLoginExample {
  security.jaas.MyLogin required debug=true;
  };
```

jaas.config ' un bir JAAS oturum açma modülünü tanımlamasıyla ilgili bilgi edinmek için Telemetri kanalı JAAS yapılandırması konusuna bakın.

7. Add a telemetry channel using the **Yeni telemetri kanalı** wizard in WebSphere MQ Explorer, configuring the channel to require JAAS authentication. MyLoginExample Stanza 'ya başvurun.

For example, adapt the information you type into the wizard from this stanza in the mqxr_win.properties file. Linux içinde çalışıyorsanız, dosya mqxr_unix.propertiesolarak adlandırılırr. teletext properties dosyasını doğrudan düzenlemeyin; sihirbazı kullanın.

```
com.ibm.mq.MQXR.channel/JAASMCAUser: \
com.ibm.mq.MQXR.Port=1884;\
com.ibm.mq.MQXR.JAASConfig=MyLoginExample;\
com.ibm.mq.MQXR.UserName=Admin;\
com.ibm.mq.MQXR.StartWithMQXRService=true
```

Not: Telemetri kanalı değiştirgelerinden herhangi birini değiştirirseniz ya da security.jaas.Mylogin sınıfını değiştirirseniz, telemetri (MQXR) hizmetini durdurup yeniden başlatmanız gerekir. Yalnızca, hizmeti yeniden başlattığınızda değişikliklerin yürürlüğe girmesi için gerekli değişiklikleri yapın.

8. Make a copy of PubSync.java in the com.ibm.mq.id package, and name the copy PubSyncJAAS.java.

com.ibm.mq.id paketinde PubSync.java yaratılmasına ilişkin adımlar için bkz. “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 .

9. Set the MqttConnectOptions.username and MqttConnectOptions.password in PubSyncJAAS.java program, and pass MqttConnectOptions as a parameter of MqttClient.connect.

```
MqttConnectOptions conOptions = new MqttConnectOptions();
conOptions.setUserName(Example.username);
conOptions.setPassword(Example.password);
client.connect(conOptions);
```

PubSyncJAAS.Java , Example.java' da ayarlanan değişmezleri kullanır.

10. Set Example.TCPAddress to the socket address of the telemetry channel you have configured to use the JAAS configuration, MyLoginExample. Örneğin, kapı numarası olarak 1884 seçeneğini kullanın.
11. İstemcinin oturum açmasını görmek ve kabul edilmek ya da reddedilmek için PubSyncJAAS ' u birkaç kez çalıştırın.

Oturum açma girişimi her reddedildiğinde bir kural dışı durum oluşur.

Sonuçlar

Şekil 103 sayfa 479 , PubSyncJAAS.Java seçeneğini iki kez tıklatın. Günlük kayıtları Şekil 104 sayfa 479içinde gösterilir.

```
Waiting for up to 10 seconds for publication of "Hello World Fri Jun 04 08:31:05 BST 2010" with
QoS = 1
On topic "MQTT Example" for client instance: "Admin_61c57a18_4bf7_40d" on address tcp://
localhost:1884"
With username "Admin" and password "Password"
Client exception caught
Client is not connected (32104)
    at
com.ibm.micro.client.mqttv3.internal.ExceptionHelper.createMqttException(ExceptionHelper.java:33
)
    at com.ibm.micro.client.mqttv3.internal.ClientComms.internalSend(ClientComms.java:88)
    at com.ibm.micro.client.mqttv3.internal.ClientComms.sendNoWait(ClientComms.java:105)
    at com.ibm.micro.client.mqttv3.MqttTopic.publish(MqttTopic.java:68)
    at com.ibm.mq.id.PubSync.main(PubSync.java:24)
```

```
Waiting for up to 10 seconds for publication of "Hello World Fri Jun 04 08:31:40 BST 2010" with
QoS = 1
On topic "MQTT Example" for client instance: "Admin_1d1599a0_50f5_4ea" on address tcp://
localhost:1884"
With username "Admin" and password "Password"
Delivery token "1731749688" has been received: true
```

Şekil 103. PubSyncJAAS.java' dan konsol çıkışı

The log file `MyLogin.log` is stored in *`WMQ Data directory`*; for example, `C:\IBM\MQ\Data\MyLogin.log`:

```
Called JAASLogin.login at Hello World Fri Jun 04 08:31:05 BST 2010
Username: "Admin", Password: "Password" loggedIn: false
Called JAASLogin.login at Hello World Fri Jun 04 08:31:40 BST 2010
Username: "Admin", Password: "Password" loggedIn: true
```

Şekil 104. MyLogin.log

Örnekler

The italicized code in Şekil 105 sayfa 480 is the modification to the sample `JAASLoginModule.java`.

```
package security.jaas;
import java.io.FileWriter;
import java.io.PrintWriter;

public class JAASLogin implements javax.security.auth.spi.LoginModule {
  private javax.security.auth.Subject                  subject;
  private javax.security.auth.callback.CallbackHandler callbackHandler;
  JAASPrincipal                                        principal;
  boolean                                              loggedIn = false;
  public void initialize(javax.security.auth.Subject subject,
      javax.security.auth.callback.CallbackHandler callbackHandler,
      java.util.Map<String, ?> sharedState, java.util.Map<String, ?> options) {
    this.subject = subject;
    this.callbackHandler = callbackHandler;
  }
  public boolean login() throws javax.security.auth.login.LoginException {
    try {
      javax.security.auth.callback.Callback[] callbacks = new
javax.security.auth.callback.Callback[2];
      callbacks[0] = new javax.security.auth.callback.NameCallback(
          "NameCallback");
      callbacks[1] = new javax.security.auth.callback.PasswordCallback(
          "PasswordCallback", false);

      callbackHandler.handle(callbacks);
      String username = ((javax.security.auth.callback.NameCallback) callbacks[0])
          .getName();
      char[] password = ((javax.security.auth.callback.PasswordCallback) callbacks[1])
          .getPassword();
      // login half the users randomly
      PrintWriter pw = new PrintWriter(new FileWriter(System
          .getProperty("user.dir")
          + "\\mylogin.log", true));
      pw.println("Called JAASLogin.login at "
          + System.getProperty("publication", "Hello World "
              + String.format("%tc", System.currentTimeMillis())));
      if (Math.random() < 0.5)
        loggedIn = true;
      pw.println("Username: \"" + username + "\", Password: \""
          + String.valueOf(password) + "\" loggedIn: " + loggedIn);
      pw.close();
      if (!loggedIn)
        throw new javax.security.auth.login.FailedLoginException("Login failed");
      principal = new JAASPrincipal(username);
    } catch (java.io.IOException exception) {
      throw new javax.security.auth.login.LoginException(exception.toString());
    } catch (javax.security.auth.callback.UnsupportedCallbackException exception) {
      throw new javax.security.auth.login.LoginException(exception.toString());
    }
    return loggedIn;
  }
  public boolean abort() throws javax.security.auth.login.LoginException {
    logout();
    return true;
  }
  public boolean commit() throws javax.security.auth.login.LoginException {
    if (loggedIn) {
      if (!subject.getPrincipals().contains(principal))
        subject.getPrincipals().add(principal);
    }
    return true;
  }
  public boolean logout() throws javax.security.auth.login.LoginException {
    subject.getPrincipals().remove(principal);
    principal = null;
    loggedIn = false;
    return true;
  }
}
```

Şekil 105. MyLogin.java

Şekil 106 sayfa 481 is the sample code JAASLoginPrincipal.java, copied into the package security.jaas. JAASLoginPrincipal ' in amacı, MyLoginarabiriminde başarıyla oturum açmış olan kullanıcıların kaydını tutmak için java.security.Principal arabirimini uygulamasıdır.

```
package security.jaas;
public class JAASPrincipal implements java.security.Principal,
    java.io.Serializable {
  private static final long serialVersionUID = 1L;
  String                    name;
  public JAASPrincipal(String name) {
    this.name = name;
  }
  public String getName() {
    return name;
  }
  public String toString() {
    return (name);
  }
  public boolean equals(Object object) {
    if (object != null && object instanceof JAASPrincipal
        && name.equals(((JAASPrincipal) object).getName()))
      return true;
    else
      return false;
  }
  public int hashCode() {
    return name.hashCode();
  }
}
```

Şekil 106. JAASLoginPrincipal.java

The code in PubSync.java that is modified to add a `kullanıcı adı` and `parola` is italicized in Şekil 107 sayfa 481.

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class PubSyncSSL {
  public static void main(String[] args) {
    try {
      MqttClient client = new MqttClient(Example.SSLAddress, Example.clientId);
      MqttTopic topic = client.getTopic(Example.topicString);
      MqttMessage message = new MqttMessage(Example.publication.getBytes());
      message.setQos(Example.QoS);
      MqttConnectOptions conOptions = new MqttConnectOptions();
      conOptions.setUserName(Example.username);
      conOptions.setPassword(Example.password);
      client.connect(conOptions);
      System.out.println("Waiting for up to " + Example.sleepTimeout / 1000
              + " seconds for publication of \"" + message.toString()
              + "\" with QoS = " + message.getQos());
      System.out.println("On topic \"" + topic.getName() + "\" for client instance: \""
              + client.getClientId() + "\" on address " + client.getServerURI() + "\"");
      System.out.println("With username \"" + conOptions.getUserName()
              + "\" and password \"" + String.valueOf(conOptions.getPassword()) + "\"");
      MqttDeliveryToken token = topic.publish(message);
      token.waitForCompletion(Example.sleepTimeout);
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" has been received: " + token.isComplete());
      client.disconnect();
    } catch (Exception e) {
      System.out.println("Client exception caught");
      e.printStackTrace();
    }
  }
}
```

Şekil 107. PubSyncJAAS.Java

Modify the constants in Example.java to match your configuration. Bu örneğe ilişkin SSL ayarlarını dikkate almayın.

```
package com.ibm.mq.id;
import java.util.Properties;
import java.util.UUID;
public final class Example {
  public static final String          TCPAddress =
    System.getProperty("TCPAddress", "tcp://localhost:1883");
  public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");
  public static final String            username =
    System.getProperty("username", System.getProperty("user.name"));
  public static final char []           password =
    System.getProperty("password", "Password").toCharArray();
  public static String                  clientId =
  String.format("%-23.23s", username  + "_" +
    System.getProperty("clientId",
    (UUID.randomUUID().toString())).trim()).replace('-', '_');
  public static final String         topicString =
    System.getProperty("topicString", "MQTT Example");
  public static final String         publication =
    System.getProperty("publication", "Hello World " +
  String.format("%tc", System.currentTimeMillis()));
  public static final int         quiesceTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final int           sleepTimeout =
    Integer.parseInt(System.getProperty("timeout", "10000"));
  public static final boolean       cleanSession =
    Boolean.parseBoolean(System.getProperty("cleanSession", "false"));
  public static final int                   QoS =
    Integer.parseInt(System.getProperty("QoS", "1"));
  public static final boolean           retained =
    Boolean.parseBoolean(System.getProperty("retained", "false"));
  public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore",
        "C:\\IBM\\MQ\\Data\\ClientKeyStore.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword",
        "password");
    properties.setProperty("com.ibm.ssl.trustStore",
        "C:\\IBM\\MQ\\Data\\ClientTrustStore.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType",
        "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword",
        "password");
    return properties;
  }
}
```

Şekil 108. Example.java

Kendinden onaylı sertifikalar kullanılarak SSL telemetri bağlantısının doğrulanması

SSL bağlantısını doğrulamak için **Keytool** kullanılarak oluşturulan kendinden onaylı sertifikalar kullanın. Telemetri kanalını ya da telemetri kanalını ve ona bağlanan müşterileri doğrulama seçeneğiniz var. Bağlantıda akan iletiler şifrelenir.

Başlamadan önce

Do the task, “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 before you start, to get PubSync.java working with an unsecured TCP/IP connection. Bu görevde, PubSync.java ' u bir SSL bağlantısıyla çalışacak şekilde değiştirdiniz.

Bu görev hakkında

Görevdeki adımlar, programlama alıştırıcı olarak yazılır. Bir üretim ortamında gerçek kimlik doğrulaması gerçekleştirmek için yordamı uyarlamanız gerekir.

Görev Windows için yazılmıştır. Linuxiçin dizin yollarını değiştirin.

Yordam

1. Do the task, “SSL kullanmak için PubSync.java dosyasını değiştirme” sayfa 483, to modify PubSync.java to use SSL.
2. Telemetri kanalını yapılandırın ve SSL ' yi kullanmak için anahtar depolarını oluşturun.

   Yalnızca telemetri kanalından ya da ona bağlanan kanal ve istemcilerin kimliğini doğrulayın:

   - SSL ile bağlantı kurmak için “Telemetri kanalının doğrulanması” sayfa 484görevini yapın, telemetri kanalını doğrulayın.
   - Do the task, “Telemetri kanalının ve müşterilerin kimlik doğrulaması” sayfa 485, to connect with SSL, authenticating the telemetry channel and clients that connect to it.
3. Telemetri kanalı yapılandırmalarında yapılan değişiklikleri almak için telemetri (MQXR) hizmetini durdurup yeniden başlatın.
4. Yapılanışın çalışıp çalışmayacagına ilişkin bilgi için istemci programını çalıştırın.

SSL kullanmak için PubSync.java dosyasını değiştirme

Bir telemetri kanalına SSL kullanarak bağlanmak için ilk yayınlayıcı programı örneğini değiştirin. Değiştirilen program tarafından kullanılan SSL özelliklerini ayarlayın.

Başlamadan önce

You are assumed to have installed the MQTT v3 client jar files, Javadoc, Eclipse, configured telemetry channels and coded and run PubSync.java before performing this task. You have an Eclipse workspace that includes a running version of PubSync.java.

Bu görev hakkında

The task uses the publisher client, PubSync.java, you created in “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 as a base. SSL kullanmak için yalnızca küçük değişiklikler gereklidir; bkz. Şekil 109 sayfa 484 ve Şekil 110 sayfa 484.

Yordam

1. Make a copy of `PubSync.java` in the `com.ibm.mq.id` package, and name the copy `PubSyncSSL.java`.

   `com.ibm.mq.id` paketinde PubSync.java yaratılmasına ilişkin adımlar için bkz. “Java kullanan ilk MQ Telemetry Transport Publisher uygulamasının yaratılması” sayfa 455 .
2. SSL yapılandırması için kullanmak üzere yapılandırdığınız telemetri kanalının yuva adresini `Example.SSLAddress` olarak ayarlayın.
3. İstemci oluşturucusunun yuva adresi parametresini `Example.SSLAddress`kullanacak şekilde değiştirin.

   ```
   MqttClient client = new MqttClient(Example.SSLAddress, Example.clientId);
   ```
4. Set the `MqttConnectOptions.SSLProperties` in `PubSyncSSL.java`, and pass `MqttConnectOptions` as a parameter of `MqttClient.connect`.

   ```
   MqttConnectOptions conOptions = new MqttConnectOptions();
   conOptions.setSSLProperties(Example.getSSLSettings());
   client.connect(conOptions);
   ```

   `Example.java`içinde ayarlanan sabitleri kullanarak, italik kodu `PubSyncSSL.Java` içine bakın.

Örnekler

SSL ' yi eklemek için PubSync.java üzerinde yapılan değişiklikler, Şekil 109 sayfa 484 içinde italik bir şekilde gösterilir.

```
package com.ibm.mq.id;
import com.ibm.micro.client.mqttv3.*;
public class PubSyncSSL {
  public static void main(String[] args) {
    try {
      MqttClient client = new MqttClient(Example.SSLAddress, Example.clientId);
      MqttTopic topic = client.getTopic(Example.topicString);
      MqttMessage message = new MqttMessage(Example.publication.getBytes());
      message.setQos(Example.QoS);
      MqttConnectOptions conOptions = new MqttConnectOptions();
      conOptions.setSSLProperties(Example.getSSLSettings());
      client.connect(conOptions);
      System.out.println("Waiting for up to " + Example.sleepTimeout / 1000
              + " seconds for publication of \"" + message.toString()
              + "\" with QoS = " + message.getQos());
      System.out.println("On topic \"" + topic.getName() + "\" for client instance: \""
              + client.getClientId() + "\" on address " + client.getServerURI() + "\"");
      System.out.println("SSL Properties" + conOptions.getSSLProperties());
      MqttDeliveryToken token = topic.publish(message);
      token.waitForCompletion(Example.sleepTimeout);
      System.out.println("Delivery token \"" + token.hashCode()
          + "\" has been received: " + token.isComplete());
      client.disconnect();
    } catch (Exception e) {
      System.out.println("Client exception caught");
      e.printStackTrace();
    }
  }
}
```

Şekil 109. PubSyncSSL.Java

Example.java üzerinde yapılan değişiklikler Şekil 110 sayfa 484içinde gösterilir.

```
 public static final String          SSLAddress =
    System.getProperty("SSLAddress", "ssl://localhost:8883");

 public static final Properties getSSLSettings() {
    final Properties properties = new Properties();
    properties.setProperty("com.ibm.ssl.keyStore", "C:\\Certificates\\SSClientKey.jks");
    properties.setProperty("com.ibm.ssl.keyStoreType", "JKS");
    properties.setProperty("com.ibm.ssl.keyStorePassword", "password");
    properties.setProperty("com.ibm.ssl.trustStore", "C:\\Certificates\\SSClientTrust.jks");
    properties.setProperty("com.ibm.ssl.trustStoreType", "JKS");
    properties.setProperty("com.ibm.ssl.trustStorePassword", "password");
    return properties;
```

Şekil 110. Example.java' da yapılan değişiklikler

Telemetri kanalının doğrulanması

Müşteriler, kanalda akan iletilerin içeriğini şifrelemek ve bir müşterinin doğru telemetri kanalına bağlanmasını sağlamak için telemetri kanalını doğrulamaktadır. Sunucu, istemcinin kimliğini doğrulamıyor.

Bu görev hakkında

Kendinden onaylı sertifikalar yaratmak ve bunları yönetmek için farklı anahtar deposu düzenleyicilerinden birini kullanabilirsiniz. Görev, JRE ' nin bir parçası olan **keytool** komut satırı komutunu kullanır. Anahtar depolarına göz atmak ve anahtarlar oluşturmak için WebSphere MQ ile birlikte verilen **iKeyman**grafik kullanıcı arabirimi aracını kullanabilirsiniz. Launch **iKeyman** using the command **strmqikm**.

Yordam

1. **Yeni telemetri kanalı** sihirbazını kullanarak SSL bağlantısı gerektiren bir telemetri kanalı oluşturun `SSLSSOptClients` . Kanal anonim istemcileri kabul eder.

   Kanal yapılandırmanızı aşağıdaki yapılandırmadan uyarlayın. teletext properties dosyasını doğrudan düzenlemeyin; sihirbazı kullanın.

   ```
   com.ibm.mq.MQXR.channel/SSLSSOptClients: \
   com.ibm.mq.MQXR.Port=8883;\
   com.ibm.mq.MQXR.Backlog=4096;\
   com.ibm.mq.MQXR.KeyFileName=C:\\Certificates\\SSServerOptKey.jks;\
   com.ibm.mq.MQXR.PassPhrase=password;\
   com.ibm.mq.MQXR.ClientAuth=OPTIONAL;\
   com.ibm.mq.MQXR.UserName=Admin;\
   com.ibm.mq.MQXR.StartWithMQXRService=true
   ```

2. Telemetri kanalının kimliğini doğrulamak için istemci için anahtarları oluşturun.

   a) Yeni bir anahtar depodaki telemetri kanalı için kendinden onaylı bir anahtar çifti oluşturun, `SSServerOptKey.jks`:

   ```
   Keytool -genkey -noprompt -alias SSServerPrivate
           -dname "CN=mqttserver.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore SSServerOptKey.jks -storepass password -keypass password
   ```

   b) `-rfc` seçeneğini kullanarak, genel sertifikasını bir ASCII dosyası olarak dışa aktarın:

   ```
   Keytool -export -noprompt -alias SSServerPrivate -file SSServerPublic.cer
           -keystore SSServerOptKey.jks -storepass password -rfc
   ```

   Görevi pencerelerde çalıştırıyorsanız, içeriğini incelemek için `SSServerPublic.cer` simgesini çift tıklatın.

   c) Genel sertifikayı yeni bir istemci güvenli deposuna (SSClientTrust.jks) içe aktarın:

   ```
   Keytool -import -noprompt -alias SSServerPublic -file SSServerPublic.cer
           -keystore SSClientTrust.jks -storepass password
   ```

   d) Boş bir istemci anahtar deposu (`SSClientKey.jks`) oluşturun.

   Keytool boş bir anahtar deposu yaratmak için bir komut bulunmaz. İki seçeneğiniz vardır:

   i) **strmqikm**komutunu çalıştırın ve bir anahtar deposu (`SSClientKey.jks`) oluşturun, ancak herhangi bir anahtar eklemeyin.

   ii) “Telemetri kanalının ve müşterilerin kimlik doğrulaması” sayfa 485adımında 3a adımını gerçekleştirin, ancak anahtarları henüz kullanmayın.

Telemetri kanalının ve müşterilerin kimlik doğrulaması

Müşteriler, telemetri kanalını doğrular ve telemetri kanalı, bu kanala bağlanan istemcilerin kimliklerini doğrular. Kanalda akan iletiler şifrelenir.

Bu görev hakkında

Kendinden onaylı sertifikalar yaratmak ve bunları yönetmek için farklı anahtar deposu düzenleyicilerinden birini kullanabilirsiniz. Görev, JRE ' nin bir parçası olan **keytool** komut satırı komutunu kullanır. Anahtar depolarına göz atmak ve anahtarlar oluşturmak için WebSphere MQ ile birlikte verilen **iKeyman**grafik kullanıcı arabirimi aracını kullanabilirsiniz. Launch **iKeyman** using the command **strmqikm**.

Telemetri kanalı, görev için farklı bir anahtar deposuyla yapılandırılıyor, “Telemetri kanalının doğrulanması” sayfa 484. Anahtar deposuna anahtar eklemek için aynı anahtar deposunu kullanabilir ve “2” sayfa 486 adımını atlayabilirsiniz.

Yordam

1. **Yeni telemetri kanalı** sihirbazını kullanarak SSL bağlantısı gerektiren bir telemetri kanalı oluşturun SSLSSReqClients . Kanal yalnızca kimliği doğrulanmış istemcileri kabul eder.

   Kanal yapılandırmanızı aşağıdaki yapılandırma gösterisinden uyarlayın:

   ```
   com.ibm.mq.MQXR.channel/SSLSSReqClients: \
   com.ibm.mq.MQXR.Port=8884;\
   com.ibm.mq.MQXR.Backlog=4096;\
   com.ibm.mq.MQXR.KeyFileName=C:\\Certificates\\SSServerReqKey.jks;\
   com.ibm.mq.MQXR.PassPhrase=password;\
   com.ibm.mq.MQXR.ClientAuth=REQUIRED;\
   com.ibm.mq.MQXR.UserName=Admin;\
   com.ibm.mq.MQXR.StartWithMQXRService=true
   ```

2. Telemetri kanalının kimliğini doğrulamak için istemci için anahtarları oluşturun.

   a) Yeni bir anahtar depodaki telemetri kanalı için kendinden onaylı bir anahtar çifti oluşturun, SSServerReqKey.jks:

   ```
   Keytool -genkey -noprompt -alias SSServerPrivate
           -dname "CN=mqttserver.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore SSServerReqKey.jks -storepass password -keypass password
   ```

   b) -rfc seçeneğini kullanarak, genel sertifikasını bir ASCII dosyası olarak dışa aktarın:

   ```
   Keytool -export -noprompt -alias SSServerPrivate -file SSServerPublic.cer
           -keystore SSServerReqKey.jks -storepass password -rfc
   ```

   Görevi pencerelerde çalıştırıyorsanız, içeriğini incelemek için SSServerPublic.cer simgesini çift tıklatın.

   c) Genel sertifikayı yeni bir istemci güvenli deposuna (SSClientTrust.jks) içe aktarın:

   ```
   Keytool -import -noprompt -alias SSServerPublic -file SSServerPublic.cer
           -keystore SSClientTrust.jks -storepass password
   ```

3. Bir müşterinin kimliğini doğrulamak için telemetri kanalı için anahtarları oluşturun.

   a) Yeni bir anahtar depoda istemci için kendinden onaylı bir anahtar çifti oluşturun, SSClientKey.jks:

   ```
   Keytool -genkey -noprompt -alias SSClientPrivate
           -dname "CN=mqttclient.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore SSClientKey.jks -storepass password -keypass password
   ```

   b) -rfc seçeneğini kullanarak, genel sertifikasını bir ASCII dosyası olarak dışa aktarın:

   ```
   Keytool -export -noprompt -alias SSClientPrivate -file SSClientPublic.cer
           -keystore SSClientKey.jks -storepass password -rfc
   ```

   Görevi pencerelerde çalıştırıyorsanız, içeriğini incelemek için SSClientPublic.cer simgesini çift tıklatın.

   c) Genel sertifikayı sunucu anahtar deposuna (SSServerReqKey.jks) içe aktarın:

   ```
   Keytool -import -noprompt -alias SSClientPublic -file SSClientPublic.cer
           -keystore SSServerReqKey.jks -storepass password
   ```

   Telemetri kanalları, hem özel anahtarlar hem de güvenilir sertifikalar için aynı mağazeyi kullanır.

Sertifika zinciri kullanarak bir SSL telemetri bağlantısının doğrulanması

Bir SSL bağlantısının kimliğini doğrulamak için, bir sertifika yetkilisinden ya da kendi sertifika yordamınızı uygulamaktan alınan imzalı sertifikaları kullanın. Telemetri kanalını ya da telemetri kanalını ve ona bağlanan müşterileri doğrulama seçeneğiniz var. Bağlantıda akan iletiler şifrelenir.

Başlamadan önce

Do the task, “Kendinden onaylı sertifikalar kullanılarak SSL telemetri bağlantısının doğrulanması” sayfa 482 before you start, to get PubSyncSSL.Java working with a secured TCP/IP connection using self-signed certificates.

Bu görev hakkında

In this task, modify the tasks “Telemetri kanalının doğrulanması” sayfa 484, and “Telemetri kanalının ve müşterilerin kimlik doğrulaması” sayfa 485 in “Kendinden onaylı sertifikalar kullanılarak SSL telemetri bağlantısının doğrulanması” sayfa 482, to work with keys certified by a certificate chain.

Bu göreve ilişkin sertifikaları bir sertifika yetkilisinden alabilir ya da sertifikaları almak için http://www.openca.org/ gibi bir web sitelerini kullanabilirsiniz. Ticari sertifika yetkilileri, genellikle deneme sertifikalarını ücretsiz olarak kısa bir süre için sağlar. Bu görev, ticari olarak elde edilen sertifikalar kullanılarak sınanmıştı.

Başka bir seçenek de, https://www.openssl.org/gibi web sitelerindeki araçları kullanarak kendi sertifikasyon sürecinizi oluşturmak ve bunu kendi bilgisayarlarınızda çalıştırmak.

JRE `cacerts` güvenilir depoları bu görevde kullanılmıyor. You can use the JRE `cacerts` truststore at the client in the task, “Telemetri kanalının doğrulanması” sayfa 487, instead of using the specified truststore. The certificate chain might be signed by a well known certificate authority that already has its root certificate in the `cacerts` store at the client. Bu durumda, istemcide bir güvenilir depo belirtmeyin. İstemcide kurulu birden çok JRE varsa, doğru `cacerts` deposunu yönettiğinizden emin olun.

Yordam

1. If you have not already done so, do the task, “SSL kullanmak için PubSync.java dosyasını değiştirme” sayfa 483, to modify PubSync.java to use SSL.
2. Telemetri kanalını yapılandırın ve SSL ' yi kullanmak için anahtar depolarını oluşturun.

   Yalnızca telemetri kanalından ya da ona bağlanan kanal ve istemcilerin kimliğini doğrulayın:

   - SSL ile bağlantı kurmak için “Telemetri kanalının doğrulanması” sayfa 487görevini yapın, telemetri kanalını doğrulayın.
   - Do the task, “Telemetri kanalının ve müşterilerin kimlik doğrulaması” sayfa 489, to connect with SSL, authenticating the telemetry channel and clients that connect to it.
3. Telemetri kanalı yapılandırmalarında yapılan değişiklikleri almak için telemetri (MQXR) hizmetini durdurup yeniden başlatın.
4. Yapılanışın çalışıp çalışmayacagına ilişkin bilgi için istemci programını çalıştırın.

Telemetri kanalının doğrulanması

Müşteriler, kanalda akan mesajların içeriğini şifrelemek ve bir müşterinin doğru telemetri kanalına bağlanmasını sağlamak için telemetri kanalını doğrular. Sunucu, istemcinin kimliğini doğrulamıyor.

Bu görev hakkında

Sertifikaları yaratmak ve yönetmek için farklı anahtar deposu düzenleyicilerinden birini kullanabilirsiniz. Görev, JRE ' nin bir parçası olan **`keytool`** komut satırı komutunu kullanır. Anahtar depolarına göz atmak ve anahtarlar oluşturmak için WebSphere MQ ile birlikte verilen **`iKeyman`**grafik kullanıcı arabirimi aracını kullanabilirsiniz. Launch **`iKeyman`** using the command **`strmqikm`**.

Yordam

1. **Yeni telemetri kanalı** sihirbazını kullanarak SSL bağlantısı gerektiren bir telemetri kanalı oluşturun `SSLCAOptClients` . Kanal anonim istemcileri kabul eder.

   Kanal yapılandırmanızı aşağıdaki yapılandırmadan uyarlayın. teletext properties dosyasını doğrudan düzenlemeyin; sihirbazı kullanın.

```
com.ibm.mq.MQXR.channel/SSLCAOptClients: \
com.ibm.mq.MQXR.Port=8885;\
com.ibm.mq.MQXR.Backlog=4096;\
com.ibm.mq.MQXR.KeyFileName=C:\\Certificates\\CAServerOptKey.jks;\
com.ibm.mq.MQXR.PassPhrase=password;\
com.ibm.mq.MQXR.ClientAuth=OPTIONAL;\
com.ibm.mq.MQXR.UserName=Admin;\
com.ibm.mq.MQXR.StartWithMQXRService=true
```

2. Telemetri kanalının kimliğini doğrulamak için istemci için CA imzalı bir anahtar oluşturun.

   a) Yeni bir anahtar depodaki telemetri kanalı için kendinden onaylı bir anahtar çifti oluşturun, `SSServerOptKey.jks`:

   ```
   Keytool -genkey -noprompt -alias CAServerPrivate -keyalg RSA
           -dname "CN=mqttserverOpt.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore CAServerOptKey.jks -storepass password -keypass password
   ```

   Bazı sertifika yetkilileri bunu gerektirdiğinden, anahtar algoritması RSA olarak ayarlıdır. Sertifikana ilişkin ortak ad benzersiz olmalıdır, bazı sertifika yetkilileri aynı ortak adlara sahip anahtarları yayınlamaz.

   b) ASCII dosyası olarak bir sertifika imza isteği (CSR) yaratır

   ```
   Keytool -certreq -noprompt -alias CAServer -file CAServerOptKey.csr
           -dname "CN=mqttserverReq.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore CAServerOptKey.jks -storepass password -keypass password
   ```

   c) Sertifika yetkilisi yazılımını çalıştırın ya da web sitelerinde oturum açın. CMR dosyası sorulduğunda `CAServerOptKey.csr` içeriğine yapıştırın.

   d) Sertifika yetkilisi, ASCII dosyaları olarak bir ya da iki sertifika ve imzalanmış bir yanıt dosyası döndürür. İçeriği iki ya da üç dosyaya yapıştırın:

   kök sertifika
   `CARoot.cer`içine yapıştır

   Orta düzey sertifika
   `CAInter.cer`içine yapıştır

   Sunucu imzalı yanıt dosyası
   `CAServerOpt.rsp`içine yapıştır

   Bu görevde JRE sertifikaları deposu kullanılmaz. Bir kök sertifika ve CA ' dan imzalanmış bir yanıt aldıysanız, aşağıdaki adımlarda kök sertifikayı ve imzalı yanıtı kullanın. Bir kök ve bir ara sertifika aldıysanız, ara sertifikayı ve imzalı yanıtı kullanın.

   e) İmzalanan sunucu yanıtını, sertifika isteğini yayınlamış olduğunuz sunucu anahtar deposuna alır.

   Yanıtın alınması, sertifika kuruluşu tarafından imzalanmış olması için kendinden onaylı sertifikayı değiştirir. Yanıtı aldıktan sonra ve aldıktan sonra anahtar depodaki sertifikaya bakarsanız, imzalayanın değişiklikleri olur. Böyle bir işlem yoksa, bir hata, anahtar yönetim aracı tarafından bildirilir. Sertifikayı kullanmadan önce, sertifikayı inceleyin ve imzalayanın artık CA ' nın olduğunu doğrulayın.

   ```
   Keytool -import -noprompt -alias CAServer -file CAServerOpt.rsp
           -keystore CAServerOptKey.jks -storepass password
   ```

   iKeymangibi bazı anahtar yönetimi yazılımında, içe aktarma yerine, yanıt dosyaları alırsınız.

   f) CA sertifikasını istemci güvenilirlik deposuna aktarın.

   CA ' dan iki sertifika aldıysanız, ara sertifikayı ya da yalnızca bir sertifika aldıysanız, kök sertifikayı içe aktarın.

   Aşağıdakilerden birini yapın:

   ```
   keytool -import -alias CAInter -file CAInter.cer
           -keystore CAClientTrust.jks -storepass password
   ```

Ya da:

```
keytool -import -alias CARoot -file CARoot.cer
        -keystore CAClientTrust.jks -storepass password
```

Telemetri kanalının ve müşterilerin kimlik doğrulaması

Müşteriler, telemetri kanalını doğrular ve telemetri kanalı, bu kanala bağlanan istemcilerin kimliklerini doğrular. Kanalda akan iletiler şifrelenir.

Bu görev hakkında

Sertifikaları yaratmak ve yönetmek için farklı anahtar deposu düzenleyicilerinden birini kullanabilirsiniz. Görev, JRE ' nin bir parçası olan **keytool** komut satırı komutunu kullanır. Anahtar depolarına göz atmak ve anahtarlar oluşturmak için WebSphere MQ ile birlikte verilen **iKeyman**grafik kullanıcı arabirimi aracını kullanabilirsiniz. Launch **iKeyman** using the command **strmqikm**.

The telemetry channel is configured with a different keystore to the one in the task, “Telemetri kanalının doğrulanması” sayfa 487. Anahtar deposuna anahtar eklemek için aynı anahtar deposunu kullanabilir ve “2” sayfa 489 adımını atlayabilirsiniz.

Yordam

1. **Yeni telemetri kanalı** sihirbazını kullanarak SSL bağlantısı gerektiren bir telemetri kanalı oluşturun `SSLCAReqClients` . Kanal yalnızca kimliği doğrulanmış istemcileri kabul eder.

   Kanal yapılandırmanızı aşağıdaki yapılandırmadan uyarlayın. teletext properties dosyasını doğrudan düzenlemeyin; sihirbazı kullanın.

   ```
   com.ibm.mq.MQXR.channel/SSLCAReqClients: \
   com.ibm.mq.MQXR.Port=8886;\
   com.ibm.mq.MQXR.Backlog=4096;\
   com.ibm.mq.MQXR.KeyFileName=C:\\Certificates\\CAServerReqKey.jks;\
   com.ibm.mq.MQXR.PassPhrase=password;\
   com.ibm.mq.MQXR.ClientAuth=REQUIRED;\
   com.ibm.mq.MQXR.UserName=Admin;\
   com.ibm.mq.MQXR.StartWithMQXRService=true
   ```

2. Telemetri kanalının kimliğini doğrulamak için istemci için CA imzalı bir anahtar oluşturun.

   a) Yeni bir anahtar depodaki telemetri kanalı için kendinden onaylı bir anahtar çifti oluşturun, `CAServerReqKey.jks`:

   ```
   Keytool -genkey -noprompt -alias CAServerPrivate -keyalg RSA
           -dname "CN=mqttserverReq.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore CAServerReqKey.jks -storepass password -keypass password
   ```

   Bazı sertifika yetkilileri bunu gerektirdiğinden, anahtar algoritması RSA olarak ayarlıdır. Sertifikana ilişkin ortak ad benzersiz olmalıdır, bazı sertifika yetkilileri aynı ortak adlara sahip anahtarları yayınlamaz.

   b) ASCII dosyası olarak bir sertifika imza isteği (CSR) yaratır

   ```
   Keytool -certreq -noprompt -alias CAServer -file CAServerReqKey.csr
           -dname "CN=mqttserverReq.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
           -keystore CAServerReqKey.jks -storepass password -keypass password
   ```

   c) Sertifika yetkilisi yazılımını çalıştırın ya da web sitelerinde oturum açın. CMR dosyası sorulduğunda `CAServerReqKey.csr` içeriğine yapıştırın.

   d) Sertifika yetkilisi, ASCII dosyaları olarak bir ya da iki sertifika ve imzalanmış bir yanıt dosyası döndürür. İçeriği iki ya da üç dosyaya yapıştırın:

   kök sertifika
   `CARoot.cer`içine yapıştır

Orta düzey sertifika
`CAInter.cer`içine yapıştır

Sunucu imzalı yanıt dosyası
`CAServerReq.rsp`içine yapıştır

Bu görevde JRE sertifikaları deposu kullanılmaz. Bir kök sertifika ve CA ' dan imzalanmış bir yanıt aldıysanız, aşağıdaki adımlarda kök sertifikayı ve imzalı yanıtı kullanın. Bir kök ve bir ara sertifika aldıysanız, ara sertifikayı ve imzalı yanıtı kullanın.

e) İmzalanan sunucu yanıtını, sertifika isteğini yayınlamış olduğunuz sunucu anahtar deposuna alır.

Yanıtın alınması, sertifika kuruluşu tarafından imzalanmış olması için kendinden onaylı sertifikayı değiştirir. Yanıtı aldıktan sonra ve aldıktan sonra anahtar depodaki sertifikaya bakarsanız, imzalayanın değişiklikleri olur. Bu işlem yoksa ve hata, anahtar yönetim aracı tarafından raporlanır. Sertifikayı kullanmadan önce, sertifikayı inceleyin ve imzalayanın artık CA ' nın olduğunu doğrulayın.

```
Keytool -import -noprompt -alias CAServer -file CAServerReq.rsp
        -keystore CAServerReqKey.jks -storepass password
```

iKeymangibi bazı anahtar yönetimi yazılımında, içe aktarma yerine, yanıt dosyaları alırsınız.

f) CA sertifikasını istemci güvenilirlik deposuna aktarın.

CA ' dan iki sertifika aldıysanız, ara sertifikayı ya da yalnızca bir sertifika aldıysanız, kök sertifikayı içe aktarın.

Aşağıdakilerden birini yapın:

```
keytool -import -alias CAInter -file CAInter.cer
        -keystore CAClientTrust.jks -storepass password
```

Ya da:

```
keytool -import -alias CARoot -file CARoot.cer
        -keystore CAClientTrust.jks -storepass password
```

3. Müşterilerin kimliklerini doğrulamak için telemetri kanalı için CA imzalı bir anahtar oluşturun.

a) Yeni bir anahtar depodaki istemciler için kendinden onaylı bir anahtar çifti oluşturun, `CAClientKey.jks`:

```
Keytool -genkey -noprompt -alias CAClientPrivate -keyalg RSA
        -dname "CN=mqttserverReq.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
        -keystore CAClientKey.jks -storepass password -keypass password
```

Bazı sertifika yetkilileri bunu gerektirdiğinden, anahtar algoritması RSA olarak ayarlıdır. Sertifikana ilişkin ortak ad benzersiz olmalıdır, bazı sertifika yetkilileri aynı ortak adlara sahip anahtarları yayınlamaz.

b) ASCII dosyası olarak bir sertifika imza isteği (CSR) yaratır

```
Keytool -certreq -noprompt -alias CAClient -file CAClientKey.csr
        -dname "CN=mqttserverReq.ibm.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
        -keystore CAClientKey.jks -storepass password -keypass password
```

c) Sertifika yetkilisi yazılımını çalıştırın ya da web sitelerinde oturum açın. CMR dosyası sorulduğunda `CAClientKey.csr` içeriğine yapıştırın.

d) Sertifika yetkilisi, ASCII dosyaları olarak bir ya da iki sertifika ve imzalanmış bir yanıt dosyası döndürür. İçeriği iki ya da üç dosyaya yapıştırın:

kök sertifika
`CARoot.cer`içine yapıştır

Orta düzey sertifika
`CAInter.cer`içine yapıştır

İstemci tarafından imzalanmış yanıt dosyası
`CAClient.rsp`içine yapıştır

Bu görevde JRE sertifikaları deposu kullanılmaz. Bir kök sertifika ve CA ' dan imzalanmış bir yanıt aldıysanız, aşağıdaki adımlarda kök sertifikayı ve imzalı yanıtı kullanın. Bir kök ve bir ara sertifika aldıysanız, ara sertifikayı ve imzalı yanıtı kullanın.

e) İmzalanan istemci yanıtını, sertifika isteğini yayınlamış olduğunuz istemci anahtar deposuna alır.

Yanıtın alınması, sertifika kuruluşu tarafından imzalanmış olması için kendinden onaylı sertifikayı değiştirir. Yanıtı aldıktan sonra ve aldıktan sonra anahtar depodaki sertifikaya bakarsanız, imzalayanın değişiklikleri olur. Bu işlem yoksa ve hata, anahtar yönetim aracı tarafından raporlanır. Sertifikayı kullanmadan önce, sertifikayı inceleyin ve imzalayanın artık CA ' nın olduğunu doğrulayın.

```
Keytool -import -noprompt -alias CAClient -file CAClient.rsp
        -keystore CAClientKey.jks -storepass password
```

iKeymangibi bazı anahtar yönetimi yazılımında, içe aktarma yerine, yanıt dosyaları alırsınız.

f) CA sertifikasını sunucu anahtar deposuna aktarın.

CA ' dan iki sertifika aldıysanız, ara sertifikayı ya da yalnızca bir sertifika aldıysanız, kök sertifikayı içe aktarın.

Aşağıdakilerden birini yapın:

```
keytool -import -alias CAInter -file CAInter.cer
        -keystore CAServerReqKey.jks -storepass password
```

Ya da:

```
keytool -import -alias CARoot -file CARoot.cer
        -keystore CAServerReqKey.jks -storepass password
```

İlk MQ Telemetry Transport Publisher uygulamasının C kullanılarak yaratılması

Bir MQTT istemcisi yayınlayıcı uygulaması yaratma adımları, öğretici program modasında açıklanmıştır. C kodunun her bir satırı açıklanır. Görevin sonunda, bir MQTT yayınlayıcısı yaratmış bulunmuyorsunuz.

Başlamadan önce

Geliştirilen istemci uygulaması, istemci MQTT v3 C istemci kitaplıklarını kullanır. Uygulama, iletileri yayınlamak için aygıtlar için WebSphere MQ Telemetry yardımcı programına bağlanır. WebSphere MQ Telemetryile iletişim kuran bir istemciye ilişkin örnek için bkz. İlk yayınlayıcınız yaratılıyor .

Bu görev hakkında

Örnek, bir yayınlama uygulamasıdır, `pubsync.c`. `pubsync.c` programı, `Hello World!` bilgi yükünün `MQTT Example`ile bir ileti yayınını yayınlar ve yayının yardımcı programa teslim edildiğini doğrulamayı bekler.

Basitlik için, kullanılan bazı işlevlerden dönüş kodları doğru tamamlama için test edilmemektedir. Üretim kodunda, programın beklendiği gibi davrandığından emin olmak için dönüş kodları denetleyebilirler. Beklenmeyen bir hata oluşursa, uygun işlem gerçekleştirilmelidir.

Bir abone `MQTT Example` olarak ayarlanarak, uygulamanın çalıştığını denetleyebilirsiniz.

İstemciyi geliştirmek, oluşturmak ve çalıştırmak için seçtiğiniz C geliştirme ortamınızı kullanın. İsterseniz, kodu doğrudan örneklerden kopyalayabilirsiniz.

Yordam

1. Yeni, boş bir kaynak dosya oluşturun, `pubsync.c`
2. Bir dosya oluşturun, `settings.h`. Şekil 2 'deki kodu dosyaya kopyalayın.

Programda kullanılan tüm parametreler `settings.` hiçinde tanımlanır. Dosyadaki değerleri değiştirerek, değerleri geçersiz kılabilirsiniz.

3. Takip eden adımlar, kodu açıklar. Adımları izleyin ya da kodu Figure 1 'den `pubsync.c` ' a kopyalayın.
4. Üstbilgi dosyasını eklemek için gerekli standart kitaplıklara ve `MQTTClient.h` ve `settings.h` dosyalarına ilişkin deyimler yer alır.

```
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "MQTTClient.h"
#include "settings.h"
```

5. `main()` işlevinin tanımlamasını başlatın.

```
int main(int argc, char* argv[])
{
```

6. Programda kullanılan yerel değişkenleri tanımlayın.

```
MQTTClient client;
MQTTClient_connectOptions conn_opts = MQTTClient_connectOptions_initializer;
MQTTClient_message pubmsg;
MQTTClient_deliveryToken token;
int rc;
```

Not: Connection options are required by the `MQTTClient_connect` function. `MQTTClient_connectOptions_initializer` , varsayılan seçenekleri içerir.

7. Bir istemci oluşturun.

```
MQTTClient_create(&client, ADDRESS, CLIENTID, MQTTCLIENT_PERSISTENCE_NONE, NULL);
```

- `& client` , yeni oluşturulan istemciye ilişkin bir tanıtıcıyı gösteren bir işarettir. Bu işlev, 0 dönüş koduyla geri döndüğünde, yeni istemciye bir tanıtıcı değeri içerir. Bu örnek, başarılı olduğunu varsayar. Üretim kodunda doğru tamamlama için hata kodunu test edin.
- ADDRESS , yardımcı program tarafından gelen istemci bağlantısı istekleri için izlenen MQTT kapısının URI 'sidir.
- `CLIENTID` , istemciyi yardımcı programla tanıtmak için kullanılan addır. Her etkin istemcinin benzersiz bir adı olmalıdır. Çalışan iki istemcide bir istemci tanıtıcısını kopyaladığınızda, her iki istemciye de bir özel durum oluşur ve bir istemci sonlandırılır. Bu ad, yardımcı program tarafından bir bağlantının kesilmesiyle ilgili olarak bir istemci tjhat bağlantısının tanınması için kullanılır. Bkz. İstemci tanıtıcısı.
- `MQTTCLIENT_PERSISTENCE_NONE` , istemci durumunun bellekta tutulduğunu ve bir sistem hatasının ortaya çıkmasının kaybolduğunu belirtir. `MQTTCLIENT_PERSISTENCE_DEFAULT` , dosya sistemine dayalı kalıcılığı belirtir; hatalara karşı koruma sağlar. Daha özel uygulamalar için, kendi kalıcılık mekanizmanızı uygulayabilmeniz için bir arabirim sağlayan `MQTTCLIENT_PERSISTENCE_USER`komutunu kullanabilirsiniz. Daha fazla ayrıntı için, `MQTTClientPersistence.` hiçin API belgelerine bakın. Kalıcılık gerekli olup olmadığı bir uygulama tasarım sorudur. Daha fazla ayrıntı için bakınız: Message persistence.
- MQTT için varsayılan yardımcı program TCP/IP kapısı 1883 'tür. Örnekte, varsayılan adres `tcp://localhost:1883`olarak ayarlıdır.
- `MQTTClient_connect` işlevini çağırıncaya kadar, hiçbir ileti işleme gerçekleşmez.

8. İstemciyi cine bağlan.

```
if ((rc = MQTTClient_connect(client, &conn_opts)) != MQTTCLIENT_SUCCESS) {
     printf("Failed to connect, return code %d\n", rc);
     exit(-1);
}
```

- `MQTTClient_connect` işlevi çağrılır, istemci tanıtıcısını ve bağlantı seçeneklerine bağımsız değişken olarak bir gösterge geçirilir.

- `MQTTClient_connect` çağrısından dönen dönüş kodu, bağlantı isteğinin başarılı olduğundan emin olmak için sınanmış.
- `MQTTClient_connect` başarısız olursa program, -1 hata koduyla biter.
- Uygulama bağlandıktan sonra, yayınlama ve abone olma işlemini başlatabilirsiniz.
- TCP/IP bağlantısının kapatılmasını önlemek için her 20 saniyede bir küçük bir "keep-canliy" (keep-canlı) iletisi gönderilir. Bu seçenek `conn_opts.keepAliveInterval`tarafından ayarlanır.
- `conn_opts.cleansession` değeri true olarak ayarlandığından, önceki bir bağlantıdan geri kalan inflight iletilerinin tamamlanması denetlenmeden oturum başlatılır. Daha fazla ayrıntı için bkz. Temizleme oturumları.
- Bağlantı için son bir irade ve ahit iletisi yaratılmadı. Daha fazla ayrıntı için bkz. Son irade ve ahit.

9. Populate the *MQTTClient_message* structure with the data to define the message payload and its attributes.

```
pubmsg.payload = PAYLOAD;
pubmsg.payloadlen = strlen(PAYLOAD);
pubmsg.qos = QOS;
pubmsg.retained = 0;
```

   - BILGI YüKü, ileti içeriğimizdir.
   - Bu örnek bir dizgi bilgi yükü kullanır, ancak MQTT bilgi yüklemeleri byte dizileridir. Bilgi yükü büyüklüğünü belirlemek için dizgi uzunluğu gereklidir.
   - Örnek, bir QoS=1 iletisi yayınlar, değeri buna göre ayarlayın.
   - İleti, yardımcı program tarafından saklanmadığı için false (0) olarak ayarlıdır. Daha fazla ayrıntı için bkz. Yayınların edinilişi.

10. İletiyi yayınlayın.

```
MQTTClient_publishMessage(client, TOPIC, &pubmsg, &token);
```

    - Yayınlama işlevi, istemciyi, konuyu ve yardımcı programa gönderilecek bilgi yükünü belirtir.
    - KONU , `settings.h` içinde `MQTT Example`olarak tanımlanır.
    - İşlev, bir *MQTTClient_deliveryToken*' e bir işaretçi iletir. Bu işaretçi, işlev döndürdüğünde iletiyi gösteren bir simgeyle doldurulur.
    - İleti şimdi MQTT istemcisine güvenli bir şekilde aktarıldı, ancak yardımcı programa aktarılmadı. İletinin QoS=1 ya da 2 değeri varsa, teslim işlemi tamamlanmadan istemcinin başarısız olması durumunda ileti yerel olarak saklanır.
    - Bu işlev, üretim kodunda doğru tamamlama için test edebildiğiniz bir hata kodu döndürür.

11. Sunucudan onay beklenmesini bekleyin.

```
rc = MQTTClient_waitForCompletion(client, token, TIMEOUT);
```

    - `pubsync.c` örneği, iletinin teslim edildiğini onaylayan sunucudan bir alındı bildirimi bekler.
    - İstemci ve simge bağımsız değişkenleri, programın tamamlanabilmek için beklediği iletiyi tanımlar.
    - TIMEOUT , programın teslimi tamamlamak için ne kadar süre bekleyeceğini belirler. C kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı oluşturma görevi, geri bildirme işlevleri kullanılarak beklemeden kabul edilen onayların nasıl alınacağını gösterir.
    - Bu işlev, üretim kodunda doğru tamamlama için test edilebilen bir hata kodu döndürür.

12. İstemcinin yardımcı programdan bağlantısını kesin.

```
MQTTClient_disconnect(client, 10000);
```

    - İstemci sunucudan bağlanıyor ve beliren iletilerin tamamlanması için bu örnekte kullanılmayan geri bildirme işlevleri (bu örnekte kullanılmadı) bekliyor.
    - İkinci bağımsız değişken, milisaniye cinsinden susturma zamanaşımını belirtir. Örnek, bağlantıyı kesmeden önce yapması gereken diğer işleri bitirmek için 10 saniyeye kadar bekler.

- Bu işlev, üretim kodunda doğru tamamlama için sınanması gereken bir hata kodu döndürür.

13. İstemci tarafından kullanılan serbest bellek belleği ve program sona erdirilsin.

```
MQTTClient_destroy(&client);
}
```

Sonuçlar

Bu istemci tarafından gönderilen yayınları görmek için, `MQTT Example` konusuna bir abone oluşturun. Daha fazla ayrıntı için bkz. C kullanarak MQ Telemetry Transport için abone oluşturma

Örnek

Şekil 1 , Procedure(Yordam) içinde açıklanan kodun tam listesidir. Şekil 2 'deki `settings.h` dosyası, `pubsync.c`' ta kullanılan varsayılan parametreleri değiştirmenize olanak tanır.

```
#include "stdio.h"
#include "stdlib.h"
#include "MQTTClient.h"
#include "settings.h"

int pubsync_main(int argc, char* argv[]) {
    MQTTClient client;
    MQTTClient_connectOptions conn_opts = MQTTClient_connectOptions_initializer;
    MQTTClient_message pubmsg;
    MQTTClient_deliveryToken token;
    int rc;

    MQTTClient_create(&client, ADDRESS, CLIENTID,  MQTTCLIENT_PERSISTENCE_NONE, NULL);
    if ((rc = MQTTClient_connect(client, &conn_opts)) != MQTTCLIENT_SUCCESS) {
          printf("Failed to connect, return code %d\n", rc);
         exit(-1);
    }
    pubmsg.payload = PAYLOAD;
    pubmsg.payloadlen = strlen(PAYLOAD);
    pubmsg.qos = QOS;
    pubmsg.retained = 0;
    MQTTClient_publishMessage(client, TOPIC, &pubmsg, &token);
    printf("Waiting for up to %d seconds for publication of %s\n"
          "on topic %s for client with ClientID: %s\n",
          TIMEOUT/1000, PAYLOAD, TOPIC, CLIENTID);
    rc = MQTTClient_waitForCompletion(client, token, TIMEOUT);
    printf("Message with delivery token %d delivered\n", token);
    MQTTClient_disconnect(client, 10000);
    MQTTClient_destroy(&client);
}
```

Şekil 111. `pubsync.c`

```
#define ADDRESS   "tcp://localhost:1883"
#define CLIENTID  "ExampleClientPub"
#define TOPIC     "MQTT Example"
#define PAYLOAD   "Hello World!"
#define QOS      1
#define TIMEOUT   10000L
```

Şekil 112. `settings.h`

C kullanarak MQ Telemetry Transport için zamanuyumsuz bir yayınlayıcı oluşturma

Bir MQTT istemcisi zamanuyumsuz yayınlayıcı uygulaması yaratma adımları, öğretici program modasında açıklanmıştır. C kodunun her bir satırı açıklanır. Görevin sonunda, bir MQTT zamanuyumsuz yayınlayıcı yaratmış olmasınız.

Bu görevde, ilk yayınlayıcı uygulamanızı değiştirmek için bir öğretici program izlemeniz gerekir. Değişiklikler, uygulamanın teslim onaylarını beklemeden yayınları göndermesine olanak sağlar. Teslim onayları, oluşturduğunuz geri bildirme işleviyle alınır.

Başlamadan önce

Geliştirilen istemci uygulaması, istemci MQTT v3 C istemci kitaplıklarını kullanır. Uygulama, iletileri yayınlamak için aygıtlar için WebSphere MQ Telemetry yardımcı programına bağlanır. WebSphere MQ Telemetryile iletişim kuran bir istemciye ilişkin örnek için bkz. İlk yayınlayıcınız yaratılıyor .

Bu görev hakkında

Örnek, bir yayınlama uygulamasıdır, `pubasync.c`. The program `pubasync.c` publishes a message with the payload `Hello World!` to the topic `MQTT Example`, without waiting for confirmation that the publication has been delivered to the daemon. The delivery acknowledgments are received in a callback function, `MQTTClient_deliveryComplete`.

Basitlik için, kullanılan bazı işlevlerden dönüş kodları doğru tamamlama için test edilmemektedir. Üretim kodunda, programın beklendiği gibi davrandığından emin olmak için dönüş kodları denetleyebilirler. Beklenmeyen bir hata oluşursa, uygun işlem gerçekleştirilmelidir.

Bir abone `MQTT Example` olarak ayarlanarak, uygulamanın çalıştığını denetleyebilirsiniz.

İstemciyi geliştirmek, oluşturmak ve çalıştırmak için seçtiğiniz C geliştirme ortamınızı kullanın.

Yordam içindeki adımlar, “İlk MQ Telemetry Transport Publisher uygulamasının C kullanılarak yaratılması” sayfa 491uygulamasından `pubsync.c` uygulamasını değiştirmektedir. İsterseniz, kodu doğrudan örneklerden kopyalayabilirsiniz.

Yordam

1. Boş bir kaynak dosya (`callback.h`) oluşturun.
2. Kodu Şekil 2 ' deki dosyaya kopyalayın.
   - `callback.h` , zamanuyumsuz istemci işlemi için gereken üç geri çağırma yöntemini bildirir.
   - Bir değişken (*deliveredtoken*) da bildirilir. Bu, ana program ve yürütmenin farklı iş parçacıklarına ilişkin geri bildirme tarafından erişilir. Bu nedenle geçici olarak bildirilmiş olur. Geri çağrıları kullanırken, ilgili değişkenlere iş parçacığı güvenli bir şekilde erişildiğinden emin olmak için dikkatli olun.
3. Boş bir kaynak dosya (`callback.c`) oluşturun.
4. Kodu Şekil 3 ' te dosyaya kopyalayın.
   - `callback.c` , istemci tarafından zamanuyumsuz işlem için kullanılan üç geri çağırma yöntemini uygular, `delivered`, `msgarrvd`ve `connlost`.
5. Add an include statement for `callback.h` after the other includes in `pubasync.c`.

   ```
   #include "callback.h"
   ```

6. `pubsync.c` içeriğini yeni bir dosyaya (`pubasync.c.`) kopyala
7. `pubasync.c`' ta `MQTTClient_connect` işlev çağrısından hemen önce, istemciye ilişkin geri çağırma yöntemlerini ayarlayın.

   ```
   MQTTClient_setCallbacks(client, NULL, connlost, msgarrvd, delivered);
   ```

   - Üç geri bildirme işlevi belirtmeniz gerekir. Bu işlevler `callback.c`içinde uygulanır.
   - `MQTTClient_messageArrived` , eşleşen bir abonelik nedeniyle istemciye bir ileti gönderildiğinde çağrılır. Alınan ileti istemci uygulaması tarafından başarıyla alındıysa, bu değer true değerini döndürmelidir. False değeri döndürülmesi, istemciye uygulamanızın iletiyi almakta bir sorun olduğunu gösterir.
   - `MQTTClient_connectionLost` is called when the client loses its connection to the server.
   - `MQTTClient_deliveryComplete` is called when a QoS1 or QoS2 message has arrived and been acknowledged by the server. QoS0 iletileri için çağrılmaz. In the example, this function saves the token from the delivered message in *teslim edilişi simgesi* to indicate that a message has arrived.

- İstemcinin sunucuyla bağlantısı kesilirken`MQTTClient_setCallbacks` çağrılmalıdır.
- İkinci bağımsız değişken, geri bildirme işlevlerine bağlamsal bilgileri geçirmenizi sağlar. Bu, örnekte kullanılmıyorsa, NULL (boş değer) olarak ayarlıdır.

8. `MQTTClient_publishMessage`çağrısından hemen önce *deliveredtoken*öğesini temizleyin. `MQTTClient_deliveryComplete` , bir simge alındığında *deliveredtoken* öğesini ayarlar.

```
deliveredtoken = 0;
```

9. Remove the `MQTTClient_waitForCompletion` call and the `printf` statement following it and replace with a loop waiting for a match of the original token and the token received in the callback.

```
while(deliveredtoken != token);
```

Bu bir örnektir ve üretim kodu tasarımında konaklaması gereken durumlardan bir sayıyla başa çıkmaz. Bu durumlar şunlardır:

- Teslim alma işlemi tamamlanmaması durumunda, zamanaşımı uygulanabilir
- Birden çok ileti girilebilir. Örnek program yalnızca bir teslim simgesinin bir kerede denetlenmesine izin verir.

10. İstemcinin yardımcı programdan bağlantısını kesin.

```
MQTTClient_disconnect(client, 10000);
```

- İstemci sunucudan bağlantısını keser ve ileti ışığı iletilerinin tamamlanmak üzere geri bildirme işlevlerini bekleyeceğini bekler.
- İkinci bağımsız değişken, milisaniye cinsinden susturma zamanaşımını belirtir. Örnek, bağlantıyı kesmeden önce yapması gereken diğer işleri bitirmek için 10 saniyeye kadar bekler.
- Bu işlev, üretim kodunda doğru tamamlama için sınanması gereken bir hata kodu döndürür.

11. İstemci tarafından kullanılan serbest bellek belleği ve program sona erdirilsin.

```
MQTTClient_destroy(&client);
}
```

Sonuçlar

Bu istemci tarafından gönderilen yayını görmek için, `MQTT Example` konusuna bir abone oluşturun. Daha fazla ayrıntı için bkz. MQ Telemetry Transport için abone oluşturma

Örnek

`pubasync.c`, `callbacks.c` ve `callbacks.h` , Yordam' da açıklanan kodun tam listeleridir.

```
#include "stdio.h"
#include "stdlib.h"
#include "MQTTClient.h"
#include "settings.h"
#include "callback.h"

int main(int argc, char* argv[]) {
    MQTTClient client;
    MQTTClient_connectOptions conn_opts = MQTTClient_connectOptions_initializer;
    MQTTClient_message pubmsg;
    MQTTClient_deliveryToken token;
    int rc;

    MQTTClient_create(&client, ADDRESS, CLIENTID, MQTTCLIENT_PERSISTENCE_NONE, NULL);
    MQTTClient_setCallbacks(client, NULL, connlost, msgarrvd, delivered);
    if ((rc = MQTTClient_connect(client, &conn_opts)) != MQTTCLIENT_SUCCESS) {
        printf("Failed to connect, return code %d\n", rc);
        exit(-1);
    }
    pubmsg.payload = PAYLOAD;
    pubmsg.payloadlen = strlen(PAYLOAD);
    pubmsg.qos = QOS;
    pubmsg.retained = 0;
    deliveredtoken = 0;
    MQTTClient_publishMessage(client, TOPIC, &pubmsg, &token);
    printf("Waiting for publication of %s\n"
           "on topic %s for client with ClientID: %s\n", PAYLOAD, TOPIC, CLIENTID);
    while(deliveredtoken != token);
    MQTTClient_disconnect(client, 10000);
    MQTTClient_destroy(&client);
}
```

Şekil 113. pubasync.c

```
MQTTClient_deliveryComplete delivered;
MQTTClient_messageArrived msgarrvd;
MQTTClient_connectionLost connlost;

extern volatile MQTTClient_deliveryToken deliveredtoken;
```

Şekil 114. callback.h

```
#include "MQTTClient.h"

volatile MQTTClient_deliveryToken deliveredtoken;

void delivered(void *context, MQTTClient_deliveryToken dt)
{
    printf("Message with token value %d delivery confirmed\n", dt);
    deliveredtoken = dt;
}

int msgarrvd(void *context, char *topicName, int topicLen, MQTTClient_message *message)
{
    int i;
    char* payloadptr;

    printf("Message arrived\n");
    printf("     topic: %s\n", topicName);
    printf("   message: ");

    payloadptr = message->payload;
    for(i=0; i<message->payloadlen; i++) {
        putchar(*payloadptr++);
    }
    putchar('\n');
    MQTTClient_freeMessage(&message);
    free(topicName);
    return 1;
}

void connlost(void *context, char *cause)
{
    printf("\nConnection lost\n");
    printf("     cause: %s\n", cause);
}
```

Şekil 115. `callback.c`

```
#define ADDRESS    "tcp://localhost:1883"
#define CLIENTID   "ExampleClientPub"
#define TOPIC      "MQTT Example"
#define PAYLOAD    "Hello World!"
#define QOS        1
#define TIMEOUT    10000L
```

Şekil 116. `settings.h`

C kullanarak MQ Telemetry Transport için abone yaratılması

Bir MQTT istemcisi abone uygulaması yaratma adımları, öğretici program modasında açıklanmıştır. C kodunun her bir satırı açıklanır. Görevin sonunda, bir MQTT abonesi yaratmış olmasınız.

Başlamadan önce

Geliştirilen istemci uygulaması, istemci MQTT v3 C istemci kitaplıklarını kullanır. Uygulama, iletileri yayınlamak için aygıtlar için WebSphere MQ Telemetry yardımcı programına bağlanır. WebSphere MQ Telemetryile iletişim kuran bir istemciye ilişkin örnek için bkz. İlk yayınlayıcınız yaratılıyor .

Bu görev hakkında

Örnek, bir abone uygulaması (`subscribe.c`). The program `subscribe.c` subscribes to the topic `MQTT Example` and waits for publications that match the subscription until the user ends the program.

Abone, bir konuya ilişkin abonelik yaratır ve abonelik konularıyla eşleşen iletileri bekler. İstemci bağlantısı kesildiğinde ve daha önce istemci tarafından yaratılan bir aboneliğle eşleşen iletiler, istemci yeniden bağlandığında alınabilir. Aygıtlar için WebSphere MQ telemetry (MQXR) hizmeti ya da cini, istemci tanıtıcısı tarafından önceden bağlanmış bir istemciyi tanır. Ek bilgi için İstemci tanıtıcısıbaşlıklı konuya bakın. `MQTTClient_connectOptions.cleansession` Boole özniteliği, önceden gönderilen yayınların alınıp alınmayacağını denetler. Daha fazla ayrıntı için bkz. “Temizleme oturumları” sayfa 507.

Basitlik için, kullanılan bazı işlevlerden dönüş kodları doğru tamamlama için test edilmemektedir. Üretim kodunda, programın beklendiği gibi davrandığından emin olmak için dönüş kodları denetleyebilirler. Beklenmeyen bir hata oluşursa, uygun işlem gerçekleştirilebilir.

You can use the previously described publish example programs to send matching publications to WebSphere MQ Telemetry daemon for devices. Diğer bir seçenek olarak, istemciyi bir WebSphere MQ Telemetry kanalına bağlamak istiyorsanız, `MQTT Example` konusunda test yayınlarını yaratmak için WebSphere MQ gezginini kullanın.

The instructions in Yordam assume that you have already created `callback.c`, `callback.h`, and `settings.h` files in one of the earlier tasks.

İstemciyi geliştirmek, oluşturmak ve çalıştırmak için seçtiğiniz C geliştirme ortamınızı kullanın. İsterseniz, kodu doğrudan örneklerden kopyalayabilirsiniz.

Yordam

1. Bu örnek için `settings.h` ' nin bir kopyasını oluşturun ve *CLIENTID* tanımlama deyimini aşağıdakine çevirin:

   ```
   #define CLIENTID "ExampleClientSub"
   ```

   - Aynı tanıtıcıya sahip iki istemci tek bir sunucuya bağlanmaya çalışırsa, bunlardan biri zorla çıkarılır. Genellikle, yeni bağlanma girişimi başarılı olur ve eski bağlantının bağlantısı kesilir.
   - `ClientID` ' nın değiştirilmesi, bu aboneye ileti göndermek için önceden geliştirilmiş yayınlama örneklerini kullanmanızı sağlar.
2. Boş bir kaynak dosya (`subscribe.c`) oluşturun.
3. Takip eden adımlar, kodu açıklar. Adımları izleyin ya da kodu Şekil 117 sayfa 502 ' tan `subscribe.c`dosyasına kopyalayın.
4. Üstbilgi dosyasını eklemek için gerekli standart kitaplıklara ve `MQTTClient.h` ve `settings.h` dosyalarına ilişkin deyimler yer alır.

   ```
   #include "stdio.h"
   #include "stdlib.h"
   #include "MQTTClient.h"
   #include "settings.h"
   ```
5. `main()` işlevinin tanımlamasını başlatın.

   ```
   int main(int argc, char* argv[]) {
   ```
6. Programda kullanılan yerel değişkenleri tanımlayın.

   ```
   MQTTClient client;
   MQTTClient_connectOptions conn_opts = MQTTClient_connectOptions_initializer;
   MQTTClient_deliveryToken token;
   int rc;
   ```

   Connection options are required by the `MQTTClient_connect` function. `MQTTClient_connectOptions_initializer` , varsayılan seçenekleri içerir.
7. Bir istemci oluşturun.

   ```
   MQTTClient_create(&client, ADDRESS, CLIENTID, MQTTCLIENT_PERSISTENCE_NONE, NULL);
   ```

   - `& client` , yeni oluşturulan istemciye ilişkin bir tanıtıcıyı gösteren bir işarettir. Bu işlev, 0 dönüş koduyla geri döndüğünde, işaretçi yeni istemciye bir tanıtıcı değeri içerir. Bu örnek, başarılı olduğunu varsayar. Üretim kodunda doğru tamamlama için hata kodu test edilebilir.
   - ADDRESS , yardımcı program tarafından gelen istemci bağlantısı istekleri için izlenen MQTT kapısının URI 'sidir.
   - `CLIENTID` , istemciyi yardımcı programla tanıtmak için kullanılan addır. Her etkin istemcinin benzersiz bir adı olmalıdır. Çalışan iki istemcide bir istemci tanıtıcısını kopyaladığınızda, her iki

istemciye de bir özel durum oluşur ve bir istemci sonlandırılır. Bu ad, bir istemcinin bağlantının kesilmesiyle yeniden bağlantı kurmasını tanımak için yardımcı program tarafından kullanılır. Bkz. İstemci tanıtıcısı.

- MQTTCLIENT_PERSISTENCE_NONE , istemci durumunun bellekta tutulduğunu ve bir sistem hatasının ortaya çıkmasının kaybolduğunu belirtir. MQTTCLIENT_PERSISTENCE#_DEFAULT , başarısızlıklara karşı koruma sağlayan, dosya sistemi tabanlı kalıcılığı belirtir. Daha özel uygulamalar için, kendi kalıcılık mekanizmanızı uygulayabilmeniz için bir arabirim sağlayan MQTTCLIENT_PERSISTENCE_USERkomutunu kullanabilirsiniz. Kalıcılık gerekli olup olmadığı bir uygulama tasarım sorudur. Daha fazla ayrıntı için bakınız: Message persistence.
- MQTT için varsayılan yardımcı program TCP/IP kapısı 1883 'tür. Örnekte, varsayılan adres `tcp://localhost:1883`olarak ayarlıdır.
- `MQTTClient_connect` işlevini çağırıncaya kadar, hiçbir ileti işleme gerçekleşmez.

8. İstemciyi yardımcı programa bağlan

```
if ((rc = MQTTClient_connect(client, &conn_opts)) != MQTTCLIENT_SUCCESS) {
    printf("Failed to connect, return code %d\n", rc);
    exit(-1);
}
```

   - `MQTTClient_connect` işlevi çağrılır, istemci tanıtıcısını ve bağlantı seçeneklerine bağımsız değişken olarak bir gösterge geçirilir.
   - `MQTTClient_connect` çağrısından dönen dönüş kodu, bağlantı isteğinin başarılı olduğundan emin olmak için sınanmış.
   - Bağlantı çağrısı başarısız olursa, program, -1 hata koduyla biter.
   - Uygulama bağlandıktan sonra yayınlamaya ve abone olarak başlatılabiliyor.
   - TCP/IP bağlantısının kapatılmasını önlemek için her 20 saniyede bir küçük bir "keep-canliy" (keep-canlı) iletisi gönderilir. Bu seçenek `conn_opts.keepAliveInterval`tarafından ayarlanır.
   - *conn_opts.cleansession* değeri true olarak ayarlandığından, önceki bir bağlantıdan geri kalan inflight iletilerinin tamamlanması denetlenmeden oturum başlatılır. Daha fazla ayrıntı için bkz. Temizleme oturumları.
   - Bağlantı için son bir irade ve ahit iletisi yaratılmadı. Daha fazla ayrıntı için bkz. Son irade ve ahit

9. Konuya abone olun.

```
MQTTClient_subscribe(client, TOPIC, QOS);
```

   - İstemci uygulamasını seçilen konuya abone olmak için `MQTTClient_subscribe` işlevini kullanın. Konu adı genel arama karakterleri içerebilir. Daha fazla ayrıntı için bkz. “Topic strings and topic filters in MQTT clients” sayfa 521.
   - QoS ayarı, bu aboneye gönderilen iletilere uygulanan hizmet kalitesi üst sınırını belirler. Sunucu, bu ayarın alt değerine ve özgün iletiye ilişkin QoS ayarına ileti gönderir.
   - Bu işlev, üretim kodunda doğru tamamlama için test edilebilen bir hata kodu döndürür.

10. Kullanıcı klavyeden bir 'Q' karakteri girinceye kadar döngüde bekleyin.

```
do {
    ch = getchar();
    } while(ch!='Q' && ch != 'q');
```

    Program artık iletilerin ulaşmasını bekliyor. Bu örnekte, `MQTTClient_messageArrived`geri bildirme işlevinde tüm ileti işleme gerçekleşir. Daha fazla ayrıntı için bkz. “İletileri alma” sayfa 501.

11. İstemcinin yardımcı programdan bağlantısını kesin.

```
MQTTClient_disconnect(client, 10000);
```

    - İstemci sunucudan bağlanıyor ve beliren iletilerin tamamlanması için bu örnekte kullanılmayan geri bildirme işlevleri (bu örnekte kullanılmadı) bekliyor.

- İkinci bağımsız değişken, milisaniye cinsinden susturma zamanaşımını belirtir. Bu örnek, bağlantıyı kesmeden önce gerçekleştirmesi gereken diğer işleri bitirmek için en çok 10 saniye bekler.
- Bu işlev, üretim kodunda doğru tamamlama için test edilebilen bir hata kodu döndürür.

12. İstemci tarafından kullanılan serbest bellek belleği ve program sona erdirilsin.

```
MQTTClient_destroy(&client);
}
```

İletileri alma

Bu görev hakkında

İletiler sunucudan geldiğinde, `MQTTClient_messageArrived` işlevi başlatılır. Takip eden adımlar, kodu açıklar.

Yordam

1. Geri bildirme işlevi tanımlamasını başlatın. Bu tanımlama, `MQTTClient_messageArrived` işlev şablonunda eşleşmelidir.

```
int msgarrvd(void *context, char *topicName, int topicLen, MQTTClient_message *message) {
```

   - `bağlam` işlevi, `MQTTClient_setCallbacks` işlevi çağrıldığında istemci kitaplığına geçirilen bağlama erişim sağlar. Bu işlev, örnekte kullanılmamaktadır.
   - `topicName` , alınan iletinin yayınlandığı konu ile ilgili bir işarettir. Joker karakter kullanarak abone olmuşsanız, bu parametre ileti için kullanılan belirli konuyu tanımlar.
   - `topicLen` , konu dizgisinin uzunluğıdır. Bu seçenek, konu dizgilerinde BOŞ (NULL) karakter yerleştirmesi gereken kullanıcılar için sağlanır.
   - `ileti` , ileti bilgi yükünü ve özniteliklerini içeren `MQTTClient_message` yapısına ilişkin bir göstergedir.

2. Kullanılan yerel değişkenleri tanımlayın.

```
int i;
char* payloadptr;
```

   Bu değişkenler, üzerinde yineleme yaparak bilgi yükünü yazdırma örneğinde kullanılır.

3. İletinin görüntülenmesi, iletinin bilgi yükü ve bilgi yükü

```
printf("Message arrived\n");
printf("     topic: %s\n",topicName);
printf("   message: ");
payloadptr = message->payload;
for(i=0; i<message->payloadlen; i++){
    putchar(*payloadptr++);
    }
putchar('\n');
```

   - Bu örnek, alınan bilgi yükünün yazdırılabilir karakterlerden oluşan bir sıra olduğunu varsayar.
   - MQTT bilgi yükü bir bayt dizisidir. Bu uygulama, onların anlamlarını yorumlamaktan sorumlu olur.

4. İletiyi saklamak için kullanılan bellek serbest.

```
MQTTClient_freeMessage(&message);
MQTTClient_free(topicName);
```

   - Örnekte, tüm ileti işleme geri bildirme işlevinde yer alır.
   - Geri bildirme işlevlerinin kısa olduğundan emin olun ve en kısa sürede, çağıran iş parçacığa geri dönüş denetimi sağlayın.
   - Bu ileti göstergesi, programın ana bölümünde işlenmek üzere geçirilir.

- İşleme tamamlandığında ana programın ileti tarafından kullanılan belleği serbest bir şekilde serbest olması gerekir. `MQTTClient_freeMessage()` , *MQTTClient_message* yapısını tutmak için kullanılan iki bellek blokunu ve iletiyi sisteme geri döndüren bir kolaylık işlevidir. `topicName` ' e ayrılan bellek, ayrı olarak gösterildiği biçimde serbest bırakılmalıdır.

5. Geri çağırma iletiyi başarıyla işlediğinde true değerini döndürür

```
    return 1;
}
```

- Doğru bir değer döndürülmesi, istemci kitaplığının iletiyi başarıyla teslim edildiği gibi değerlendirebileceğini gösterir.
- Geri bildirme işlevi iletiyi doğru olarak işleyemiyorsa, false değeri döndürülür. Örneğin, geri çağırım ana program için işlenecek bir kuyruğa ileti yerleştiriyorsa ve kuyruk dolu olduğunda, false değeri döndürülmesi uygun olur.
- QoS1 ve QoS2 iletileri için, yanlış bir değer döndürülmesi, iletinin teslim edilmediğini ve teslim edilmesi için daha fazla girişimde bulunduğunu gösterir.

Örnek kod

```
#include "stdio.h"
#include "stdlib.h"
#include "MQTTClient.h"
#include "settings.h"
#include "callback.h"

int main(int argc, char* argv[]) {
    MQTTClient client;
    MQTTClient_connectOptions conn_opts = MQTTClient_connectOptions_initializer;
    int rc;
    int ch;

    MQTTClient_create(&client, ADDRESS, CLIENTID, MQTTCLIENT_PERSISTENCE_NONE, NULL);

    MQTTClient_setCallbacks(client, NULL, connlost, msgarrvd, delivered);

    if ((rc = MQTTClient_connect(client, &conn_opts)) != MQTTCLIENT_SUCCESS) {
        printf("Failed to connect, return code %d\n", rc);
        exit(-1);
        }
    printf("Subscribing to topic %s\nfor client %s using QoS%d\n\n"
           "Press Q<Enter> to quit\n\n", TOPIC, CLIENTID, QOS);

    MQTTClient_subscribe(client, TOPIC, QOS);
    do {
        ch = getchar();
        } while(ch!='Q' && ch != 'q');
    MQTTClient_disconnect(client, 10000);
    MQTTClient_destroy(&client);
}
```

Şekil 117. `subscriber.c`

```
#include "MQTTClient.h"

volatile MQTTClient_deliveryToken deliveredtoken;

void delivered(void *context, MQTTClient_deliveryToken dt) {
    printf("Message with token value %d delivery confirmed\n", dt);
    deliveredtoken = dt;
}

int msgarrvd(void *context, char *topicName, int topicLen, MQTTClient_message *message) {
    int i;
    char* payloadptr;

    printf("Message arrived\n");
    printf("     topic: %s\n", topicName);
    printf("   message: ");

    payloadptr = message->payload;
    for(i=0; i<message->payloadlen; i++) {
        putchar(*payloadptr++);
        }
    putchar('\n');
    MQTTClient_freeMessage(&message);
    MQTTClient_free(topicName);
    return 1;
}
void connlost(void *context, char *cause) {
    printf("\nConnection lost\n");
    printf("     cause: %s\n", cause);
}
```

Şekil 118. `callback.h`

```
#define ADDRESS    "tcp://localhost:1883"
#define CLIENTID   "ExampleClientSub"
#define TOPIC      "MQTT Example"
#define PAYLOAD    "Hello World!"
#define QOS        1
#define TIMEOUT    10000L
```

Şekil 119. `settings.h`

MQTT istemcisi programlama kavramları

The concepts described in this section help you to understand the Java, JavaScript and C client libraries for version 3.1 of the MQTT protocol. Kavramlar, istemci kitaplıklarına eşlik eden API belgelerini tamamlar.

`com.ibm.micro.client.mqttv3` , MQTT sürüm 3.1 iletişim kuralına ilişkin istemci kitaplıklarına ilişkin genel yöntemleri sağlayan sınıfları içerir. `com.ibm.micro.client.mqttv3` paketinin bir sürümü ve Java SE ve ME iletişim kuralını uygulayan eşlik eden paketler, IBM WebSphere MQ Telemetrykurulumu ile birlikte sağlanır. MQTT istemci kitaplıklarının en son sürümünü almak için (Java, JavaScript ve API belgelerini görüntülemek ya da karşıdan yüklemek için "MQTT client programming Reference"belgesine bakın.

Bir MQTT istemcisini geliştirmek ve çalıştırmak için, bu paketleri istemci aygıtında kopyaya ya da kurmanıza gerek vardır. Ayrı bir istemci yürütme ortamı kurmanıza gerek yoktur.

İstemciler için lisans koşulları, istemcilere bağladığınız sunucuyla ilişkilendirilir.

MQTT istemci kitaplıkları, MQTT protocol' un 3.1 sürümüne ilişkin başvuru somutlamalarıdır. Farklı aygıt platformlarına uygun farklı dillerde kendi müşterilerinizi uygulayabilirsiniz. Bkz. MQ Telemetry Transport biçimi ve iletişim kuralı.

API belgeleri, istemcinin bağlandığı MQTT sunucusu hakkında herhangi bir varsayımda yer vermez. İstemcinin davranışı, farklı sunuculara bağlanıldığında biraz farklı olabilir. The descriptions that follow describe the behavior of the client when connected to the IBM WebSphere MQ telemetry service.

MQTT istemci uygulamalarında geri çağrılar ve eşitleme

MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Geri Çağırmalar

`MqttCallback` arabiriminde üç geri arama yöntemi vardır; örneğin, Callback.javaiçinde örnek bir somutlama vardır.

`connectionLost(java.lang.Throwable cause)`

`connectionLost` , bir iletişim hatası bağlantının kesimine yol açınca çağrılır. Sunucu bağlantıyı, bağlantı kurulduktan sonra sunucudaki bir hatanın sonucu olarak düşerse de çağrılır. Sunucu hataları kuyruk yöneticisi hata günlüğüne kaydedilir. Sunucu, istemciye bağlantıyı atar ve istemci `MqttCallback.connectionLost`' i çağırır.

İstemci uygulamasıyla aynı iş parçacığıdaki kural dışı durumlar olarak verilen uzak hatalar, `MqttClient.connect`' ten kural dışı durumlardır. Errors detected by the server after the connection is established are reported back to the `MqttCallback.connectionLost` callback method as `throwables`.

`connectionLost` ile sonuçlanan tipik sunucu hataları, yetki hatalarıdır. Örneğin, telemetri sunucusu, konu üzerinde yayınlama yetkisi olmayan bir istemci adına bir konu üzerinde yayınlama girişiminde bulunuyor. Bir `MQCC_FAIL` durum kodunun telemetri sunucusuna döndürülmesi sonucunda ortaya çıkan herhangi bir şey bağlantının atılma durumuyla sonuçlanabilir.

`deliveryComplete(MqttDeliveryToken token)`

`deliveryComplete` , MQTT istemcisi tarafından istemci uygulamasına bir teslim belirteci geçirmesi için çağrılır; bkz. “Teslim simgeleri” sayfa 510. Geri çağırma, teslim simgesini kullanarak yayınlanan iletiye `token.getMessage`yöntemiyle erişebilir.

Uygulama geri çağrısı, `deliveryComplete` yöntemi tarafından çağrıldıktan sonra MQTT istemcisine denetimi geri döndürdüğünde, teslim işlemi tamamlanır. Until delivery is completed, messages with QoS 1 or 2 are retained by the persistence class.

`deliveryComplete` çağrısı, uygulama ile kalıcılık sınıfı arasındaki eşitleme noktasıdır. `deliveryComplete` yöntemi, aynı ileti için hiçbir zaman iki kez çağrılamamaktadır.

Uygulama geri çağrısı `deliveryComplete` 'dan MQTT istemcisine geri döndüğünde, istemci QoS 1 ya da 2olan iletiler için `MqttClientPersistence.remove` ' i çağırır. `MqttClientPersistence.remove` , yayınlanan iletinin yerel olarak saklanan kopyasını siler.

Bir hareket işleme perspektifinden `deliveryComplete` çağrısı, teslimi kesinleten tek aşamalı bir işlemdir. If processing fails during the callback, on restart of the client `MqttClientPersistence.remove` is called again to delete the local copy of the published message. Geri arama yeniden çağrılmaz. Geri bildirme olanağını, bir teslim edilen ileti günlüğünü saklamak için kullanıyorsanız, günlüğü MQTT istemcisiyle uyumlulaştıramazsınız. If you want to store a log reliably, then update the log in the `MqttClientPersistence` class.

Teslim simgesi ve iletisine, ana uygulama iş parçacığı ve MQTT istemcisi tarafından başvuruluyor. MQTT istemcisi, teslim işlemi tamamlandığında `MqttMessage` nesnesini ve istemci bağlantısını kestiğinde teslim belirteci nesnesini kayıttan kaldırır. The `MqttMessage` object can be garbage collected after delivery is completed if the client application dereferences it. Teslim simgesi, oturumun bağlantısı kesildikten sonra çöp toplanabilir.

Bir ileti yayınlandıktan sonra `MqttDeliveryToken` ve `MqttMessage` özniteliklerini elde edebilirsiniz. İleti yayınlandıktan sonra herhangi bir `MqttMessage` özniteliğini ayarlama girişiminde bulunursanız, sonuç tanımsız olur.

İstemci, aynı `ClientIdentifier`ile önceki oturuma yeniden bağlanıyorsa, MQTT istemcisi teslim onaylarını işlemeye devam eder; bkz. “Temizleme oturumları” sayfa 507. MQTT istemci uygulaması, önceki oturum için `MqttClient.CleanSession` öğesini `false` olarak ayarlayıp yeni oturumda `false` olarak ayarlamalıdır. MQTT istemcisi, bekleyen teslimatlar için yeni oturum için yeni teslim simgeleri ve ileti nesneleri yaratır. `MqttClientPersistence` sınıfını kullanan

nesneleri kurtarır. Uygulama istemcisinin eski teslim simgeleri ve iletilerine yönelik başvuruları varsa, bu istemcilerin başvuruları kaldırılır. Uygulama geri çağrısı, önceki oturumda başlatılan ve bu oturumda tamamlanan tüm teslimatlar için yeni oturumda çağrılır.

Uygulama geri çağrısı, uygulama istemcisi bağlandıktan sonra, bekleyen bir teslim tamamlandığında çağrılır. Before the application client connects, it can retrieve pending deliveries using the `MqttClient.getPendingDeliveryTokens` method.

İstemci uygulamasının özgün olarak yayınlanan ileti nesnesini ve bilgi yükü bayt dizisini oluşturduğunu fark edin. MQTT istemcisi bu nesnelere gönderme yapar. The message object returned by the delivery token in the method `token.getMessage` is not necessarily the same message object created by the client. Yeni bir MQTT istemcisi eşgörünümü teslim simgesini yeniden yaratacaksa, `MqttClientPersistence` sınıfı `MqttMessage` nesnesini yeniden yaratır. For consistency `token.getMessage` returns `null` if `token.isCompleted` is `true`, regardless of whether the message object was created by the application client or the `MqttClientPersistence` class.

`messageArrived(MqttTopic topic, MqttMessage message)`

`messageArrived` , bir abonelik konuyla eşleşen istemci için bir yayın geldiğinde çağrılır. konu , abonelik süzgecinin değil, yayın konudur. Süzgeç joker karakterler içeriyorsa, bu ikisi farklı olabilir.

Konu, istemci tarafından yaratılan birden çok abonelikle eşleşiyorsa, istemci yayının birden çok kopyasını alır. Bir istemci, aynı zamanda abone olduğu bir konuya yayınlarsa, kendi yayınının bir kopyasını alır.

If a message is sent with a QoS of 1 or 2, the message is stored by the `MqttClientPersistence` class before the MQTT client calls `messageArrived`. `messageArrived` behaves like `deliveryComplete`: it is only called once for a publication, and the local copy of the publication is removed by `MqttClientPersistence.remove` when `messageArrived` returns to the MQTT client. The MQTT client drops its references to the topic and message when `messageArrived` returns to the MQTT client. Uygulama istemcisi nesnelere ilişkin bir başvuruya tutmadıysa, konu ve ileti nesneleri çöp toplanır.

Geri çağrılar, threading ve istemci uygulaması eşitlemesi

MQTT istemcisi, ana uygulama iş parçacığıdaki ayrı bir iş parçacığıda bir geri çağırma yöntemi çağırır. İstemci uygulaması, geri çağırma için bir iş parçacığı oluşturmaz; bu, MQTT istemcisi tarafından yaratılır.

MQTT istemcisi geri çağırma yöntemlerini uyumlulaştırır. Geri bildirme yönteminin yalnızca bir kerede tek bir eşgörünümü çalışır. Eşitleme, yayınların teslim edildiği bir nesnenin güncellenmesini kolaylaştırır. `MqttCallback.deliveryComplete` 'un bir eşgörünümü bir kerede çalışır ve bu nedenle, daha fazla eşitleme olmadan tally' yi güncelleştirmek güvenlidir. Aynı zamanda bir kerede tek bir yayının gelmesi de geçerli olur. `messageArrived` yöntemindeki kodunuz, bir nesneyi uyumlulaştırmadan güncelleyebilir. Tally ya da güncellenmekte olan nesnede başka bir iş parçacığıysa, tally ya da object nesnesini eşitleyin.

Teslim belirteci, ana uygulama iş parçacığı ile bir yayının teslimi arasında bir eşitleme mekanizması sağlar. `token.waitForCompletion` yöntemi, belirli bir yayının teslimi tamamlanıncaya kadar bekler ya da isteğe bağlı bir zamanaşımı süresi doluncaya kadar bekler. Bir yayını aynı anda işlemek için `token.waitForCompletion` ' yi birkaç basit yolla kullanabilirsiniz:

1. Yayının teslimi tamamlanıncaya kadar uygulama istemcisini duraklatmak için Şekil 88 sayfa 460' e bakın.
2. `MqttCallback.deliveryComplete` yöntemiyle eşitlemek için. Yalnızca `MqttCallback.deliveryComplete` , MQTT Client 'a geri dönünce `token.waitForCompletion` devam eder. Using this mechanism you can synchronize running code in `MqttCallback.deliveryComplete` before code runs in the main application thread.

Ya her yayının teslim edilmesini beklemeden yayınlamak isteseniz, ancak tüm yayınlar teslim edildiğinde onay almak isteseniz? Tek bir iş parçacığıda yayınlarsanız, gönderilecek son yayın da son teslim edilecek son yayın olur.

Sunucuya gönderilen isteklerin eşitlenmesi

Çizelge 70 sayfa 506 , sunucuya bir istek gönderen MQTT Java istemcisine ilişkin yöntemleri açıklar. Uygulama istemcisi belirsiz bir zamanaşımı ayarlamazsa, istemci sunucu için süresiz olarak beklemez. İstemci askıda kalırsa, bu bir uygulama programlama sorunu ya da MQTT istemcisinden bir hata.

Çizelge 70. Sunucuya gelen isteklerle sonuçlanan yöntemlerin eşzamanlama davranışı

Yöntem	Eşitleme	Zamanaşımı aralığı
`MqttClient.Connect`	Sunucuyla bağlantı kurulması için bekler.	Varsayılan olarak 30 saniye ya da bir parametre tarafından ayarlandığı gibi bir kural dışı durum yayınlıyor.
`MqttClient.Disconnect`	MQTT istemcisinin yapması gereken işi bitirmesini ve TCP/IP oturumunun bağlantısını kesmesini bekler.	
`MqttClient.Subscribe`	Subscribe ya da UnSubscribe yönteminin tamamlanmasını bekler.	
`MqttClient.UnSubscribe`		
`MqttClient.Publish`	İsteği MQTT istemcisine ilettikten sonra uygulama iş parçacığa hemen döner.	Yok.
`MqttDeliveryToken.waitForCompletion`	Teslim simgesinin geri döndürülemesini bekler.	Belirsiz ya da parametre olarak ayarlandığı gibi.

İlgili kavramlar

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri

Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Temizleme oturumları

MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

`MqttClient.connect` yöntemini kullanarak bir MQTT istemcisi uygulaması bağladığınızda istemci, istemci tanıtıcısını ve sunucunun adresini kullanarak bağlantıyı tanımlar. Sunucu, oturum bilgilerinin sunucuya önceki bir bağlantıdan saklanıp saklanmayacağını denetler. Önceki bir oturum hala varsa ve `cleanSession=true`, istemcideki ve sunucudaki önceki oturum bilgileri temizlenir. `cleanSession=false` önceki oturuma devam ederse. Önceki oturum yoksa, yeni bir oturum başlatılır.

Not: WebSphere MQ Yöneticisi açık bir oturumu zorlamalı olarak kapatabilir ve tüm oturum bilgilerini silebilirler. İstemci oturumu `cleanSession=false`ile yeniden açarsa, yeni bir oturum başlatılır.

Yayınlar

Varsayılan `MqttConnectOptions`değerini kullanırsanız ya da istemciyi bağlamadan önce `MqttConnectOptions.cleanSession` değerini `true` olarak ayarlıyorsanız, istemci bağlandığında istemciye ilişkin bekleyen tüm yayın teslimleri kaldırılır.

Temizleme oturumu ayarının `QoS=0`ile gönderilen yayınlarda etkisi yoktur. `QoS=1` ve `QoS=2`için, `cleanSession=true` kullanılması bir yayını kaybetmeye neden olabilir.

Abonelikler

Varsayılan `MqttConnectOptions`değerini kullanırsanız ya da istemciyi bağlamadan önce `MqttConnectOptions.cleanSession` değerini `true` olarak ayarlıyorsanız, istemci bağlandığında istemciye ilişkin eski abonelikler kaldırılır. İstemcinin oturum sırasında yaptığı yeni abonelikler bağlantısı kesildiğinde kaldırılır.

If you set `MqttConnectOptions.cleanSession` to `false` before connecting, any subscriptions the client creates are added to all the subscriptions that existed for the client before it connected. İstemci bağlantısı kesildiğinde, tüm abonelikler etkin kalır.

`cleanSession` özniteliğinin abonelikleri etkilemesinin diğer bir yolu da bunu bir kalıcı öznitelik olarak algılamak olur. In its default mode, `cleanSession=true`, the client creates subscriptions and receives publications only within the scope of the session. Alternatif modda `cleanSession=false`, abonelikler dayanıklıdır. İstemci bağlanabilir ve bağlantı kesilebilir ve abonelikleri etkin kalmaya devam eder. İstemci yeniden bağlandığında, teslim edilmemiş yayınlar alır. Bağlantı bağlıyken, kendi adına etkin olan abonelikler kümesini değiştirebilir.

Bağlanmadan önce `cleanSession` kipini ayarlamanız gerekir; kip tüm oturum için geçerli olur. Ayarını değiştirmek için bağlantıyı kesmeniz ve istemciyi yeniden bağlamanız gerekir. If you change modes from

using `cleanSession=false` to `cleanSession=true`, all previous subscriptions for the client, and any publications that have not been received, are discarded.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

İstemci tanıtıcısı

İstemci tanıtıcısı, MQTT istemcisini tanımlayan 23 baytlık bir dizgidir. Her tanıtıcı, bir kerede yalnızca tek bir bağlı istemciye benzersiz olmalıdır. Tanıtıcı yalnızca, kuyruk yöneticisi adında geçerli olan karakterleri içermelidir. Bu kısıtlar içinde herhangi bir tanımlama dizilimini kullanabilirsiniz. İstemci tanıtıcılarını ayırmak için bir yordama ve seçilen tanıtıcıya sahip bir istemcinin yapılandırılmasına ilişkin bir yordama sahip olmak önemlidir.

İstemci tanıtıcısı bir MQTT sisteminin yönetiminde kullanılır. Potansiyel olarak yüz binlerce müşteriyle birlikte, belirli bir müşteriyi hızla tanımlamayı başarmanız gerekir. Örneğin, bir aygıt arızalanır ve size bir yardım masası çalan bir müşteri tarafından bildirilir. Müşteri, aygıtı nasıl tanımlar ve genellikle istemciye bağlı olan sunucu ile bu tanıma nasıl ilişkilendirilir? Her bir aygıtı bir istemci tanıtıcısına ve bir sunucuya eşleyen bir veritabanına danışmak zorunda mısınız? Aygıtın adı, hangi sunucuya bağlı olduğunu tanımlar mı? MQTT istemci bağlantılarına göz attığınızda, her bağlantı istemci tanıtıcısıyla etiketlenir. Bir istemci tanıtıcısını fiziksel bir aygıtla eşlemek için bir çizelgeye bakmanız gerekiyor mu?

İstemci tanıtıcısı, belirli bir aygıtı, kullanıcıyı mı, yoksa istemcide çalışan bir uygulamayı mı tanımlıyor? Bir müşteri arızalı bir aygıtı yeni bir aygıtla değiştirirse, yeni aygıt eski aygıtla aynı tanıtıcıya sahip olur mu? Yeni bir tanıtıcı ayırıyor musunuz? Fiziksel bir aygıtı değiştiriyorsanız, ancak aynı tanıtıcıyı alıyorsanız, olağanüstü yayınlar ve etkin abonelikler otomatik olarak yeni aygıta aktarılır.

İstemci tanıtıcılarının benzersiz olmasını nasıl sağlardınız? Benzersiz tanıtıcılar oluşturmak için bir sistem yanı sıra, istemcide tanımlayıcıyı ayarlamak için güvenilir bir işleminiz olmalıdır. Belki de istemci aygıtı, kullanıcı arabirimi olmayan bir "kara kutu" dır. Aygıtı, MAC adresini kullanmak gibi bir istemci tanıtıcısı ile üretiyor musunuz? Ya da aygıtı etkinleştirmeden önce yapılandıran yazılım kuruluşu ve yapılandırma işleminiz var mı?

Tanıtıcıyı kısa ve benzersiz tutmak için 48 bit aygıtı MAC adresinden bir istemci tanıtıcısı oluşturabilirsiniz. İletim büyüklüğü kritik bir sorun değilse, adresi denetlemek üzere geri kalan 17 baytı kullanabilirsiniz.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Teslim simgeleri

Bir istemci bir konuda yayınlandığında yeni bir teslim belirteci oluşturulur. Bir yayının teslimini izlemek için teslim simgesini ya da teslim edilinceye kadar istemci uygulamasını engellemek için kullanın.

Simge, bir `MqttDeliveryToken` nesnesidir. It is created by calling the MqttTopic.publish() method and is retained by the MQTT client until the client session is disconnected and the delivery is completed.

Simgenin normal kullanımı, teslinin tamamlanıp tamamlanmadığını kontrol etmek için kullanılır. Teslim edilinceye kadar istemci uygulamasını engelle, döndürülen simgenin kullanılması için `token.waitForCompletion`komutunu çağırın. Diğer bir seçenek olarak, bir `MqttCallBack` işleyicisi de sağlayın. When the MQTT client has received all the acknowledgments it expects as part of delivering the publication, it calls `MqttCallBack.deliveryComplete` passing the delivery token as a parameter.

Until delivery is complete, you can inspect the publication using the returned delivery token by calling `token.getMessage`.

Tamamlanan teslimatlar

Tesllerin tamamlanması zamanuyumsuz olur ve yayınla ilişkili hizmet kalitesine bağlıdır.

En çok bir kez

`QoS=0`
Delivery is complete immediately on return from `MqttTopic.publish`. `MqttCallback.deliveryComplete` hemen çağrılır.

En az bir kez

`QoS=1`
Yayın, kuyruk yöneticisinden yayınla ilgili bir alındı bildirimi alındığında tamamlanır. `MqttCallback.deliveryComplete` is called when the acknowledgment is received. İletişim yavaşsa ya da güvenilmezse, ileti `MqttCallback.deliveryComplete` çağrılmadan önce bir kereden fazla sağlanabilir.

Tam bir kez

`QoS=2`
Müşteri, yayının abonelere yayınlandığını belirten bir tamamlanma iletisi aldığında teslim edilir. `MqttCallback.deliveryComplete` is called as soon as the publication message is received. Bu, tamamlanma iletisinin beklemesini beklemez.

In rare circumstances, your client application might not return to the MQTT client from `MqttCallback.deliveryComplete` normally. You know that delivery has completed, because the `MqttCallback.deliveryComplete` was called. İstemci aynı oturumu yeniden başlattıysa, `MqttCallback.deliveryComplete` yeniden çağrılmaz.

Eksik teslim sayısı

İstemci oturumunun bağlantısı kesildikten sonra teslim tamamlanmazsa, istemciyi yeniden bağlayabilir ve teslimatı tamamlayabilirsiniz. İleti, `MqttConnectionOptions` özniteliği `false`değerine ayarlanmış bir oturumda yayınlandıysa, iletinin teslim edilmesini yalnızca tamamlayabilirsiniz.

Aynı istemci tanıtıcısını ve sunucu adresini kullanarak istemciyi yaratın ve daha sonra, `cleanSession` `MqttConnectionOptions` özniteliğini `false` ' e yeniden ayarlayarak bağlanın. `cleanSession` seçeneğini `true`olarak ayarlıyorsanız, bekleyen teslim simgeleri atılır.

Bekleyen teslimatın olup olmadığını denetlemek için `MqttClient.getPendingDeliveryTokens`' i arayarak denetleyebilirsiniz. İstemciyi bağlamadan önce `MqttClient.getPendingDeliveryTokens` ' u arayabilirsiniz.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme

MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Son irade ve vasiyet yayını

Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Son irade ve vasiyet için bir konu oluşturun. `MQTTManagement/Connections/`*`server URI`*`/`*`client identifer`*`/Lost`gibi bir konu oluşturabilirdiniz.

`MqttConnectionOptions.setWill(MqttTopic lastWillTopic, byte [] lastWillPayload, int lastWillQos, boolean lastWillRetained)` yöntemini kullanarak bir "son irade ve ahit" olarak ayarlayın.

`lastWillPayload` iletisinde bir zaman damgası yaratmayı düşünün. İstemcinin belirlenmesine ve bağlantının koşullarına yardımcı olacak diğer istemci bilgilerini de ekleyin. `MqttConnectionOptions` nesnesini `MqttClient` oluşturucusuna geçirin.

Set `lastWillQos` to 1 or 2, to make the message persistent in WebSphere MQ, and to guarantee delivery. Son kayıp bağlantı bilgilerini korumak için `lastWillRetained` , `true`olarak ayarlayın.

Bağlantının beklenmedik bir şekilde sona ermesi durumunda abonelere "son irade ve ahit" yayını gönderilir. It is sent if the connection ends without the client calling the `MqttClient.disconnect` method.

Bağlantıları izlemek için, bağlantıları kaydetmek ve programlanmış bağlantıları kaydetmek için diğer yayınlarla "last will and ahit" yayınını tamamlar.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Message persistence in MQTT clients

Publication iletileri, "en az bir kez" ya da "tam olarak bir kez" hizmet kalitesiyle gönderilirse, kalıcı kılınabilirler. İstemcide kendi kalıcılık mekanizmanızı uygulayabilir ya da istemciyle birlikte sağlanan

varsayılan kalıcılık mekanizmasını kullanabilirsiniz. Kalıcılık, istemciye ya da istemciden gönderilen yayınlar için her iki yönde de çalışır.

In MQTT, message persistence has two aspects; how the message is transferred, and whether it is queued in IBM MessageSight and IBM WebSphere MQ as a persistent message.

1. MQTT istemcisi çiftleri, hizmet kalitesiyle ileti kalıcılığı sağlar. İleti için seçtiğiniz hizmet kalitesine bağlı olarak, ileti kalıcı olarak yapılır. İleti sürekliliği, gereken hizmet kalitesini uygulamak için gereklidir.

   "En fazla bir kez" belirtirseniz, `QoS=0`, ileti yayınlanır yayınlanmaz iletiyi atar. İletinin yukarı işlenmesinde herhangi bir hata varsa, ileti yeniden gönderilmez. İstemci etkin olmaya devam ederse bile, ileti yeniden gönderilmez. `QoS=0` iletilerinin davranışı, IBM WebSphere MQ hızlı olmayan kalıcı olmayan iletiler ile aynıdır.

   Bir ileti, 1 ya da 2 değeri `QoS` olan bir istemci tarafından yayınlanırsa, kalıcı olarak yapılır. İleti yerel olarak saklanır ve "en az bir kez", `QoS=1`ya da "tam olarak bir kez", `QoS=2`, teslim garantisi vermek için artık gerekmediği durumlarda istemciden atılır.
2. Bir ileti `QoS` 1 ya da 2 olarak işaretlenirse, bu ileti IBM MessageSight ve IBM WebSphere MQ ' ta kalıcı bir ileti olarak kuyruğa alınır. `QoS=0`olarak işaretlenmişse, IBM MessageSight ve IBM WebSphere MQ içinde kalıcı olmayan bir ileti olarak kuyruğa alınır. In IBM WebSphere MQ nonpersistent messages are transferred between queue managers "exactly once", unless the message channel has the `NPMSPEED` attribute set to FAST.

Kalıcı bir yayın, istemci uygulaması tarafından alınıncaya kadar istemcide depolanır. `QoS=2`için, uygulama geri çağrısı denetimi geri döndürdüğünde, yayın istemciden atılır. `QoS=1` için, bir hata oluşursa, uygulama yayını yeniden alabilir. `QoS=0`için geri bildirme, yayını bir kereden fazla alır. Bir hata varsa ya da yayınlama sırasında istemcinin bağlantısı kesildiyse, bu yayın olmayabilir.

Bir konuya abone olduğunuzda, abonenin kalıcılık yetenekleriyle eşleşmesi için iletilerin aldığı `QoS` değerini azaltabilirsiniz. Daha yüksek bir `QoS` düzeyinde oluşturulan yayınlar, abonenin talep ettiği en yüksek `QoS` ile gönderilir.

İletilerin saklanması

küçük cihazlarda veri depolamanın uygulanması büyük bir anlaşmaya göre değişiklik gösteriyor. MQTT istemcisi tarafından yönetilen depolama alanındaki kalıcı iletilerin geçici olarak kaydedilmesi modeli çok yavaş olabilir ya da çok fazla depolama talep edebilir. Mobil aygıtlarda, mobil işletim sistemi, MQTT iletileri için ideal bir depolama hizmeti sağlayabilir.

Küçük aygıtların kısıtlamalarını yerine getirmede esneklik sağlamak için, MQTT istemcisinin iki kalıcılık arabirimi vardır. Arabirimler, kalıcı iletilerin depolanması içinde yer alan işlemleri tanımlar. Arabirimler, Javaiçin MQTT istemcisiiçin API belgelerinde açıklanmıştır. MQTT istemci kitaplıklarına ilişkin istemci API belgelerine ilişkin bağlantılar için bkz. MQTT istemci programlama başvurusu. Bir aygıta uyacak şekilde arabirimleri uygulayabilirsiniz. Java SE ' de çalışan MQTT istemcisi, dosya sisteminde kalıcı iletileri saklayan arabirimlerin varsayılan bir somutlamasını içerir. `java.io` paketini kullanır. The client also has a default implementation for Java ME, `MqttDefaultMIDPPersistence`.

Kalıcılık sınıfları

`MqttClientPersistence`
: `MqttClientPersistence` uygulamanızın bir eşgörünümünü, `MqttClient` oluşturucusunun bir parametresi olarak MQTT istemcisine geçirin. `MqttClientPersistence` parametresini `MqttClient` oluşturucudan çıkarırsanız, MQTT istemcisi kalıcı iletileri `MqttDefaultFilePersistence` ya da `MqttDefaultMIDPPersistence`sınıfını kullanarak saklar.

`MqttPersistable`
: `MqttClientPersistence` , `MqttPersistable` nesnelerini bir depolama anahtarı kullanarak alır ve yerleştirir. You must provide an implementation of `MqttPersistable` as well as the implementation of `MqttClientPersistence` if you are not using the `MqttDefaultFilePersistence` or `MqttDefaultMIDPPersistence`.

MqttDefaultFilePersistence

MQTT istemcisi `MqttDefaultFilePersistence` sınıfını sağlar. If you instantiate `MqttDefaultFilePersistence` in your client application, you can provide the directory to store persistent messages as a parameter of the `MqttDefaultFilePersistence` constructor.

Diğer bir seçenek olarak, MQTT istemcisi `MqttDefaultFilePersistence` ' yi örnek bir dizinde oluşturabilir ve dosyaları bir varsayılan dizine yerleştirebilir. Dizinin adı *`client identifier`*`-tcp` *`hostname portnumber`*. `"\", "\\", "/", ":"` ve `" "` , dizin adı dizgisinden kaldırılır.

Dizin yolu, `rcp.data`sistem özellkünün değeridir. `rcp.data` ayarlanmadıysa, yol `usr.data`sistem özelliğinin değeridir.

`rcp.data` , bir OSGi ya da Eclipse Rich Client Platform (RCP) kuruluşuyla ilişkilendirilmiş bir özeldir.

`usr.data` , uygulamanın başlatıldığı Java komutunun başlatıldığı dizindir.

MqttDefaultMIDPPersistence

`MqttDefaultMIDPPersistence` , varsayılan bir oluşturucuya sahip ve parametre yok. İletileri depolamak için `javax.microedition.rms.RecordStore` paketini kullanır.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme

MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları

MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını

Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Yayınlar

Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri

Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri

Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler

Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir.

Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Yayınlar

Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

`MqttMessage` , bilgi yükü olarak bayt dizisine sahiptir. İletilerin mümkün olduğunca küçük olmasını hedefle. MQTT iletişim kuralı tarafından izin verilen ileti uzunluğu üst sınırı 250 MB 'dir.

Tipik olarak, bir MQTT istemci programı ileti içeriğini işlemek için `java.lang.String` ya da `java.lang.StringBuffer` kullanır. Kolaylık sağlamak amacıyla, `MqttMessage` sınıfında bilgi yükünü dizgiye dönüştürecek bir `toString` yöntemi vardır. To create the byte array payload from a `java.lang.String` or `java.lang.StringBuffer`, use the `getBytes` method.

`getBytes` yöntemi, bir dizeyi, platform için varsayılan karakter kümesine dönüştürür. Varsayılan karakter kümesi genellikle `UTF-8`karakteridir. Yalnızca metin içerenMQTT yayınları genellikle `UTF-8`ile kodlanır. Varsayılan karakter kümesini geçersiz kılmak için `getBytes("UTF8")` yöntemini kullanın.

IBM WebSphere MQ' ta, bir MQTT yayını `jms-bytes` iletisi olarak alınır. İleti, `<mqtt>`ve bir `<mqps>` klasörünü içeren bir `MQRFH2` klasörü içerir. `<mqtt>` klasörü, `clientId` ve `qos`' u içerir, ancak bu içerik gelecekte değişebilir.

Bir `MqttMessage` ' de üç ek öznitelik vardır: hizmet kalitesi, alıkonulup tutulmadığına ve yinelenip yinelenmeyeceği. Yinelenen işaret, hizmet kalitesi "en az bir kez" ya da "tam olarak bir kez" olduğunda ayarlanır. İleti daha önce gönderildiyse ve MQTT istemcisi tarafından yeterince hızlı bir şekilde onaylanmadıysa, yinelenen öznitelik `true`olarak ayarlanarak ileti yeniden gönderilir.

Yayınlama

Bir MQTT istemci uygulamasında bir yayın oluşturmak için bir `MqttMessage`oluşturun. Bilgi yükünü, hizmet kalitesini ve alıkonulup tutulmadığını belirleyin ve `MqttTopic.publish(MqttMessage message)` yöntemini çağırın; `MqttDeliveryToken` iade edilir ve yayının tamamlanması da zamanuyumsuz olur.

Diğer bir seçenek olarak, MQTT istemcisi bir yayın yarattığında `MqttTopic.publish(byte [] payload, int qos, boolean retained)` yöntemindeki değiştirgelerden sizin için geçici bir ileti nesnesi yaratabilir.

Yayının "en az bir kez" ya da "tam olarak bir kez" hizmet kalitesi, `QoS=1` ya da `QoS=2`varsa, MQTT istemcisi `MqttClientPersistence` arabirimini çağırır. Uygulamaya bir teslim simgesi döndürülmeden önce iletiyi depolamak için `MqttClientPersistence` ' i çağırır.

Uygulama, `MqttDeliveryToken.waitForCompletion` yöntemini kullanarak, ileti sunucuya teslim edilinceye kadar bloke etmeyi seçebilir. Diğer bir seçenek olarak, uygulama engellemeden de devam edebilir. Yayınların engellemeden teslim olup olmadığını denetlemek istiyorsanız, `MqttCallback` istemcisini MQTT istemcisiyle gerçekleştiren bir geri çağrı sınıfı örneğini kaydedin. The MQTT client calls the `MqttCallback.deliveryComplete` method as soon as the publication has been delivered. Hizmet kalitesine bağlı olarak, teslim alma işlemi `QoS=0`için hemen hemen hemen hemen hemen hemen hemen olabilir ya da `QoS=2`için biraz zaman alabilir.

Teslimlerin tamamlandıysa yoklama yapmak için `MqttDeliveryToken.isComplete` yöntemini kullanın. `MqttDeliveryToken.isComplete` değeri `false`eise, ileti içeriğini almak için `MqttDeliveryToken.getMessage` adını arayabilirsiniz. `MqttDeliveryToken.isComplete` çağrısının sonucu `true`eise, ileti atılır ve `MqttDeliveryToken.getMessage` çağrıldığında boş değerli gösterge kural dışı durumu yayınlanır. `MqttDeliveryToken.getMessage` ile `MqttDeliveryToken.isComplete`arasında yerleşik eşitleme yok.

İstemci, bekleyen teslim simgeleri almadan önce bağlantı kesilirse, istemcinin yeni bir eşgörünümü bağlanmadan önce teslim alma simgelerini sorgulayabilir. İstemci bağlanıncaya kadar, yeni teslimatlar tamamlanmaz ve `MqttDeliveryToken.getMessage`' i aramanız güvenlidir. Hangi yayınların teslim edilmediğini öğrenmek için `MqttDeliveryToken.getMessage` yöntemini kullanın. Pending delivery tokens are discarded if you connect with `MqttConnectOptions.cleanSession` set to its default value, `true`.

abone olunması

Bir MQTT abonesine göndermek üzere yayın yaratmaktan bir kuyruk yöneticisi ya da IBM MessageSight sorumludur. Kuyruk yöneticisi, bir MQTT istemcisi tarafından yaratılan bir abonelikte konu süzgecinin yayındaki konu dizgisiyle eşleşip eşleşmediğini denetler. Eşleşme tam olarak eşleşebilir ya da eşleşme joker karakterler içerebilir. Yayın, kuyruk yöneticisi tarafından aboneye iletilmeden önce, kuyruk yöneticisi yayınla ilişkili konu özniteliklerini denetler. Bir denetim konusu nesnesinin abone olma yetkisi olup olmadığını saptamak için, Genel arama karakterleri içeren bir konu dizgisini kullanarak abone olunması başlıklı konuda açıklanan arama yordamlarından sonra gelir.

MQTT istemcisi "en az bir kez" hizmet kalitesine sahip bir yayın aldığında, yayını işlemek için `MqttCallback.messageArrived` yöntemini çağırır. Yayının hizmet kalitesi "tam olarak bir kez", `QoS=2`ise, MQTT istemcisi, iletiyi alındığında iletiyi depolamak için `MqttClientPersistence` arabirimini çağırır. Daha sonra `MqttCallback.messageArrived`' u çağırır.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir.

Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri

Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Bir yayının hizmet kalitesi, `MqttMessage`' nin bir öznitesidir. Bu, `MqttMessage.setQos`yöntemi tarafından ayarlanır.

`MqttClient.subscribe` yöntemi, bir konuyla ilgili olarak bir istemciye gönderilen yayınlara uygulanan hizmet kalitesini düşürebilir. Bir aboneye iletilen bir yayının hizmet kalitesi, yayının hizmet kalitesinden farklı olabilir. İki değerin alt değeri bir yayını iletmek için kullanılır.

En çok bir kez
`QoS=0`

İleti en çok bir kez teslim edilir ya da hiç teslim edilmez. Ağ üzerindeki teslimi onaylanmadı.
İleti saklanmaz. İstemcinin bağlantısı kesildiyse ya da sunucu başarısız olursa ileti kaybedilebilir.
`QoS=0` , aktarma için en hızlı kiptir. bazen "ateş ve unutun" denilir.
The MQTT protocol does not require servers to forward publications at `QoS=0` to a client. Sunucunun yayını aldığı süre içinde istemcinin bağlantısı kesilirse, sunucuya bağlı olarak, yayın atılabilir. Telemetri (MQXR) hizmeti, `QoS=0`ile gönderilen iletileri atmıyor. Bunlar, kalıcı olmayan iletiler olarak depolanır ve kuyruk yöneticisi durursa yalnızca atılır.

En az bir kez
`QoS=1`

`QoS=1` , varsayılan aktarma kipidir.
İleti her zaman en az bir kez teslim edilir. Gönderen bir alındı bildirimi almazsa, bir alındı bildirimi alınıncaya kadar ileti, yeniden DUP işaretiyle gönderilir. Bir sonuç alıcısı olarak aynı iletiyi birkaç kez gönderilebilir ve bunu birden çok kez işleyebilirler.
İleti, işleninceye kadar gönderene ve alıcıya yerel olarak saklanmalıdır.
İleti, iletiyi işledikten sonra alıcıdan silinir. Alıcı bir aracıysa, ileti abonelerine yayınlanır. Alıcı bir istemciyse, ileti abone uygulamasına teslim edilir. İleti silindikten sonra, alıcı gönderene bir alındı bildirimi gönderir.
İleti, alıcıdan bir alındı bildirimi aldıktan sonra göndericiden silinir.

Tam bir kez
`QoS=2`

İleti her zaman tam olarak bir kez teslim edilir.
İleti, işleninceye kadar gönderene ve alıcıya yerel olarak saklanmalıdır.
`QoS=2` , aktarım için en güvenli, ancak en yavaş kiptir. İleti göndericiden silinmeden önce, gönderen ile alıcı arasında en az iki çift iletim alır. İleti ilk iletilmeden sonra alıcıda işlenebilir.
İlk iletimler çiftinde, gönderen iletiyi iletir ve iletiyi sakladığı alıcısından alındı bildirimi alır. Gönderen bir alındı bildirimi almazsa, bir alındı bildirimi alınıncaya kadar ileti, yeniden DUP işaretiyle gönderilir.
İkinci iletimler çiftinde, gönderen, iletiyi işlemeyi tamamlayabileceğini belirtir, "PUBREL". Gönderen, "PUBREL" iletisine ilişkin bir alındı bildirimi almazsa, bir alındı bildirimi alınıncaya kadar "PUBREL" iletisi yeniden gönderilir. The sender deletes the message it saved when it receives the acknowledgment to the "PUBREL" message

İleti, iletiyi yeniden işlememesi koşuluyla, iletiyi birinci ya da ikinci aşamalarda işleyebilir. Alıcı bir aracıysa, iletiyi abonelere yayınlar. Alıcı bir istemciyse, iletiyi abone uygulamasına teslim eder. Alıcı, iletiyi işlemeyi bitirdiği bir tamamlanma iletisini gönderene geri gönderir.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Alıkonan yayınlar ve MQTT istemcileri

Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Bir konudaki yayınının alıkonulup tutulmadığını belirtmek için `MqttMessage.setRetained` yöntemini kullanın.

To delete a retained publication in IBM WebSphere MQ, run the **`CLEAR TOPICSTR`** MQSC command.

Boş değerli bilgi yükü ile bir yayın oluşturursanız, boş yayın abonelere iletilir. Diğer MQTT aracıları abonelere boş bir yayını iletemeyebilir.

Alıkonmayan bir yayını alıkonan bir yayınla ilgili bir konuya yayınlıyorsanız, alıkonan yayın bundan etkilenmez. Yürürlükteki aboneler yeni yayını alır. Yeni aboneler önce alıkonan yayını alır, sonra da yeni yayınlar alır.

Alıkonan bir yayını yarattığınızda ya da güncellediğinizde, yayını bir QoS ya da 1 ya da 2 ile gönderin. Bunu bir QoS /0 ile gönderdiğinizde, IBM WebSphere MQ , kalıcı olmayan bir alıkonacı yayın yaratır. Kuyruk yöneticisi durursa, yayın korunmaz.

Bir ölçümün en son değerini kaydetmek için alıkonan yayınları kullanın. Alıkonan konuya yeni aboneler hemen ölçümün en son değerini alır. Yayın konusuna son abone olan aboneden bu yana yeni ölçümler alınmazsa ve abone yeniden abone olursa, abone yine konuyla ilgili en son tutulan yayınını alır.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Abonelikler

Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir

ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

`MqttClient.subscribe` yöntemlerini kullanarak abonelikler oluşturun, bir ya da daha fazla konu süzgeci ve hizmet kalitesi parametreleri iletin. Hizmet kalitesi parametresi, bir iletiyi almak için abonenin kullanmak üzere hazırlamış olduğu maksimum hizmet kalitesini ayarlar. Bu istemciye gönderilen iletiler, daha yüksek hizmet kalitesiyle teslim edilemez. İleti yayınlandığında ve abonelik için belirtilen düzey olduğunda, hizmet kalitesi özgün değerin alt değerine ayarlanır. İleti almak için varsayılan hizmet kalitesi: `QoS=1`, en az bir kez.

Abonelik isteği, `QoS=1`ile birlikte gönderilir.

Publications are received by a subscriber when the MQTT client calls the `MqttCallback.messageArrived` method. `messageArrived` yöntemi, iletinin aboneye yayınlandığı konu dizesini de geçirir.

`MqttClient.unsubscribe` yöntemlerini kullanarak bir aboneliği ya da bir kümeyi ya da abonelikleri kaldırabilirsiniz.

Bir WebSphere MQ komutu, bir aboneliği kaldırabilir. List subscriptions using WebSphere MQ Explorer, or by using **runmqsc** or PCF commands. Tüm MQTT istemci abonelikleri adlandırılır. Bu formlara bir ad verilir: *`ClientIdentifier`*:*`Topic name`*

Varsayılan `MqttConnectOptions`değerini kullanırsanız ya da istemciyi bağlamadan önce `MqttConnectOptions.cleanSession` değerini `true` olarak ayarlıyorsanız, istemci bağlandığında istemciye ilişkin eski abonelikler kaldırılır. İstemcinin oturum sırasında yaptığı yeni abonelikler bağlantısı kesildiğinde kaldırılır.

If you set `MqttConnectOptions.cleanSession` to `false` before connecting, any subscriptions the client creates are added to all the subscriptions that existed for the client before it connected. İstemci bağlantısı kesildiğinde, tüm abonelikler etkin kalır.

`cleanSession` özniteliğinin abonelikleri etkilemesinin diğer bir yolu da bunu bir kalıcı öznitelik olarak algılamak olur. In its default mode, `cleanSession=true`, the client creates subscriptions and receives publications only within the scope of the session. Alternatif modda `cleanSession=false`, abonelikler dayanıklıdır. İstemci bağlanabilir ve bağlantı kesilebilir ve abonelikleri etkin kalmaya devam eder. İstemci yeniden bağlandığında, teslim edilmemiş yayınlar alır. Bağlantı bağlıyken, kendi adına etkin olan abonelikler kümesini değiştirebilir.

Bağlanmadan önce `cleanSession` kipini ayarlamanız gerekir; kip tüm oturum için geçerli olur. Ayarını değiştirmek için bağlantıyı kesmeniz ve istemciyi yeniden bağlamanız gerekir. If you change modes from using `cleanSession=false` to `cleanSession=true`, all previous subscriptions for the client, and any publications that have not been received, are discarded.

Etkin aboneliklerle eşleşen yayınlar, yayınlandığı anda istemciye gönderilir. İstemcinin bağlantısı kesildiyse, aynı istemci tanıtıcısı ve `MqttConnectOptions.cleanSession` ile `false`değerine ayarlanmış aynı sunucuya yeniden bağlanıyorsa, istemci istemciye gönderilir.

Belirli bir istemciye ilişkin abonelikler istemci tanıtıcısı tarafından tanımlanır. İstemciyi farklı bir istemci aygıtından aynı sunucuya yeniden bağlayabilir ve aynı aboneliklere devam edebilir ve teslim edilmemiş yayınlarını alabilirsiniz.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir.

Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Topic strings and topic filters in MQTT clients
Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Topic strings and topic filters in MQTT clients

Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

Konu dizgileri, abonelere yayın göndermek için kullanılır. `MqttClient.getTopic(java.lang.String topicString)` yöntemini kullanarak bir konu dizgisi yaratın.

Konu süzgeçleri konu başlıklarına abone olmak ve yayınları almak için kullanılır. Konu süzgeçleri joker karakterler içerebilir. Genel arama karakterleriyle birden çok konuya abone olabilirsiniz. Bir abonelik yöntemi kullanarak bir konu süzgeci yaratın; örneğin, `MqttClient.subscribe(java.lang.String topicFilter)`.

Konu dizgileri

Bir IBM WebSphere MQ konu dizgisinin sözdizimi, Konu Dizgileri' nde açıklanmaktadır. MQTT konu dizgilerinin sözdizimi, Javaiçin MQTT istemcisiiçin API belgelerindeki `MqttClient` sınıfında açıklanmıştır. MQTT istemci kitaplıklarına ilişkin istemci API belgelerine ilişkin bağlantılar için bkz. MQTT istemci programlama başvurusu.

Her konu dizgisinin sözdizimi hemen hemen aynı. Dört küçük fark vardır:

1. Topic strings sent to IBM WebSphere MQ by MQTT clients must follow the convention for queue manager names. Özellikle konu dizgilerinde kısa çizgi karakterleri bulunamaz.
2. Uzunluk üst sınırı farklıdır. IBM WebSphere MQ konu dizgileri 10.240 karakterle sınırlıdır. Bir MQTT istemcisi, 65535 byte 'a kadar konu dizgileri yaratabilir.
3. MQTT istemcisi tarafından yaratılan bir konu dizgisi boş değerli karakter içeremez.
4. WebSphere Message Broker 'da boş bir konu düzeyi `'...//...'` geçersizdi. Boş konu düzeyleri IBM WebSphere MQtarafından desteklenmektedir.

IBM WebSphere MQ yayınlama/abone olma gibi, `mqttv3` iletişim kuralının bir denetim konusu nesnesi kavramı yoktur. Bir konu nesnesinden ve konu dizgisinden bir konu dizgisi oluşturamazsınız. Ancak, bir konu dizgisi IBM WebSphere MQiçindeki bir yönetim konularıyla eşlenir. Yönetimle ilgili konu ile ilişkilendirilen erişim denetimi, bir yayının konuya yayınlanıp yayınlanmadığını ya da atılıp atılmadığını belirler. Abonelere iletildiğinde bir yayına uygulanan öznitelikler, denetim konularının özniteliklerinden etkilenir.

Konu süzgeçleri

Bir IBM WebSphere MQ konu süzgecinin sözdizimi, Konu tabanlı genel arama şemasıiçinde açıklanmıştır. Bir MQTT istemcisiyle oluşturabileceğiniz konu süzgeçlerinin sözdizimi, Javaiçin MQTT istemcisiiçin API belgelerindeki `MqttClient` sınıfında açıklanmıştır. MQTT istemci kitaplıklarına ilişkin istemci API belgelerine ilişkin bağlantılar için bkz. MQTT istemci programlama başvurusu.

Her konu süzgecinin sözdizimi hemen hemen aynıdır. Tek fark, farklı MQTT aracılarının bir konu süzgecini yorumlamandır. WebSphere Message Broker V6' da, çok düzeyli joker karakter yalnızca bir konu süzgecinin sonunda kullanılabilir. IBM WebSphere MQ' ta, çok düzeyli bir genel arama karakteri, konu ağacındaki herhangi bir düzeyde kullanılabilir; örneğin, `USA/#/Dutchess County`.

İlgili kavramlar

MQTT istemci uygulamalarında geri çağrılar ve eşitleme
MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

Temizleme oturumları
MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

İstemci tanıtıcısı

Teslim simgeleri

Son irade ve vasiyet yayını
Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

Message persistence in MQTT clients

Yayınlar
Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT client can create publications to send to IBM WebSphere MQ, and subscribe to topics on IBM WebSphere MQ MQ to receive publications.

MQTT istemcisi tarafından sağlanan hizmet nitelikleri
Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

Alıkonan yayınlar ve MQTT istemcileri
Alıkonan bir yayını olan bir konuya abonelik oluşturursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

Abonelikler
Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir

ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

C istemcisi programlama kavramları

MQ Telemetri Aktarımı 'nın 3.1 sürümü için C ve Java istemcisi arasındaki farklar bu konuda açıklanmaktadır. Konu, istemci kavramlarını ve C başvuru bilgilerini tamamlar.

Konu, "MQTT istemcisi programlama kavramları" sayfa 503ile aynı şekilde düzenlenmiştir. Her başlık, *WebSphere(r) MQ Telemetry Transport istemci programlama kavramlarındaki bir konuya karşılık gelir*. Bu kısımlarda, C istemcisi ile Java istemcisi arasındaki farklar açıklanmaktadır. Java yöntemleri ile C işlevleri arasındaki imzalardaki küçük farklar açıklanmaz.

C istemcisi çoğu zaman bir telemetri aygıtı ile aygıtlar için WebSphere MQ Telemetry yardımcı programı arasında hafif bir bağdaştırıcı uygulamak için kullanılır. Yardımcı program, çok hafif telemetri aygıtları ve telemetri (MQXR) hizmeti arasında ağ yoğunlaştırıcısı olarak yaygın olarak kullanılır.

Aygıtlar için WebSphere MQ Telemetry yardımcı programı aynı zamanda bir C istemcisi ve telemetri (MQXR) hizmetindeki davranışlarındaki farklar da açıklanmıştır. Yardımcı program, istemciye bağlanan istemciler için JAAS ya da SSL somutlaması sağlamaz.

`mqttclient.dll` ve `mqttclient.lib` , MQ Telemetry Transport sürüm 3.1 protokolünün C uygulaması için istemci işlevlerini içeren 32 bitlik Windows kitaplıklarıdır. 32 bit Linux kitaplıkları `libmqttclient.so` ve `libmqttclient.a`' dir. İki üstbilgi dosyası, istemci uygulamaları için gereken işlevi ve diğer bildirimleri içerir: `MQTTClient.h` ve `MQTTClientPersistence.h`. Bu dosyalar, WebSphere MQ Telemetrykuruluşu ile birlikte sağlanır.

Bir MQ Telemetry Transport istemcisini geliştirmek ve çalıştırmak için bu dosyaları istemci aygıtına kopyalamanız gerekir. WebSphere MQ istemcilerinden farklı olarak, ayrı bir istemci yürütme ortamı kurmanıza gerek yoktur.

MQ Telemetry Transport istemcilerini WebSphere MQ ve WebSphere MQ Telemetry yardımcı cinlerine bağlamayı yöneten WebSphere MQ Telemetry özelliği ile ilişkili lisans koşullarına bakın.

C istemcisi, MQ Telemetry Transport 'un 3.1 sürümünün bir başvuru uygulamasıdır. Farklı aygıt platformlarına uygun farklı dillerde kendi müşterilerinizi uygulayabilirsiniz. Ayrıntılı bilgi için MQ Telemetry Transport format and protocol belgesine bakın.

MQTT istemci tanıtıcısı

"İstemci tanıtıcısı" sayfa 508	İstemci tanıtıcısı, MQTT istemcisini tanımlayan 23 baytlık bir dizgidir. Her tanıtıcı, bir kerede yalnızca tek bir bağlı istemciye benzersiz olmalıdır. Tanıtıcı yalnızca, kuyruk yöneticisi adında geçerli olan karakterleri içermelidir. Bu kısıtlar içinde herhangi bir tanımlama dizilimini kullanabilirsiniz. İstemci tanıtıcılarını ayırmak için bir yordama ve seçilen tanıtıcıya sahip bir istemcinin yapılandırılmasına ilişkin bir yordama sahip olmak önemlidir.

- Fark yok.

Yayınlar

"Yayınlar" sayfa 515	Yayınları, bir konu dizgisiyle ilişkili `MqttMessage` eşgörünümleridir. MQTT

- Geri bildirme işlevi, "fire and forget", `QoS=0`, hizmet kalitesi ile yayınlar için çağrılmaz.

Teslim simgeleri

“Teslim simgeleri” sayfa 510	Bir istemci bir konuda yayınlandığında yeni bir teslim belirteci oluşturulur. Bir yayının teslimini izlemek için teslim simgesini ya da teslim edilinceye kadar istemci uygulamasını engellemek için kullanın.

- Teslim simgesi, `int`. Bir typedef (`MQTTClient_deliveryToken`)
- Geri bildirme işlevi, "fire and forget", `QoS=0`, hizmet kalitesi ile yayınlar için çağrılmaz.

Alıkonan yayınlar

“Alıkonan yayınlar ve MQTT istemcileri” sayfa 518	Alıkonan bir yayını olan bir konuya abonelik oluştururursanız, konuyla ilgili en son tutulan yayın, hemen size iletilir.

- Alıkonan iletiler yalnızca kalıcılık yapılandırıldıysa yardımcı programa kaydedilir; bkz. Alıkonan iletilerin ve aboneliklerin saklanması.

  WebSphere MQiçin hizmet kalitesi alıkonan bir iletinin kalıcı olarak saklanıp saklanmayacağını etkiler. Bir müşteri telemetri hizmetine bağlanırsa, "yangın ve unut" ile iletiler alıkonur, kuyruk yöneticisi kapatılırsa, `QoS=0` hizmet kalitesi atılır.

Abonelikler

“Abonelikler” sayfa 519	Bir konu süzgecini kullanarak, yayın konularına ilgi kaydetmek için abonelikler oluşturun. Bir istemci birden çok abonelik ya da genel arama karakterleri kullanan bir konu süzgeci içeren bir abonelik yaratabilir ve birden çok konuya ilgi gösterebilirler. Süzgeçlerle eşleşen konularla ilgili yayınlar istemciye gönderilir. Bir istemci bağlantısı kesilirken abonelikler etkin kalabilir. Yayınlar yeniden bağlandığında istemciye gönderilir.

- Kalıcı abonelikler, kalıcılık yapılandırılırsa yalnızca yardımcı programa kaydedilir; bkz. Alıkonan iletilerin ve aboneliklerin saklanması.
- Yayınlar eşzamanlı olarak alınabilir. `MQTTClient_receive` işlevini çağırın.

Geri çağrılar ve eşitleme

“MQTT istemci uygulamalarında geri çağrılar ve eşitleme” sayfa 504	MQTT istemci programlama modeli iş parçacıklarını kapsamlı olarak kullanır. İş parçacıkları, bir MQTT istemci uygulamasını, iletileri sunucudan ve sunucudan iletmekte olan gecikmelerden, mümkün olduğunca çok azaltan bir uygulamadan daha fazla çözer. Yayınlar, teslim simgeleri ve bağlantı kaybedilen olaylar, `MqttCallback`' u gerçekleştiren bir geri çağrı sınıfındaki yöntemlere teslim edilir.

- C istemcisinde eşitleme işlemi modüle. `MQTTClient_setCallback` ' un çağrıları, istemciyi zamanuyumsuz kipe geçirir.
- Zamanuyumlu kipte, uygulama istemcisi gönüllü olarak denetime sahip olmalıdır; bu nedenle MQTT istemcisi onayları işleyebilir ve ağı canlı tutmak için MQTT ping ping komutu yayınlayabilir. Yield control by calling `MQTTClient_receive` or `MQTTClient_yield`.

Konu dizgileri ve süzgeçleri

“Topic strings and topic filters in MQTT clients” sayfa 521	Konu dizgileri ve konu süzgeçleri yayınlamak ve abone olmak için kullanılır. MQTT istemcilerinde konu dizgilerinin ve süzgeçlerin sözdizimi büyük ölçüde IBM WebSphere MQiçindeki konu dizgileriyle aynıdır.

- The WebSphere MQ Telemetry daemon for devices handles the multi-level wildcard # differently from WebSphere MQ v7. /# must be the last two characters in the filter string for # to behave as a wildcard. WebSphere MQ v7' de, `../#/..` çok düzeyli genel arama karakterinin geçerli bir kullanışıdır. AygıtlardaWebSphere MQ Telemetry cini, çok düzeyli genel arama karakteri WebSphere MQ Broker v6ile aynı işlevi görür.

Hizmet kalitesi

“MQTT istemcisi tarafından sağlanan hizmet nitelikleri” sayfa 517	Bir MQTT istemcisi, yayınları WebSphere MQ ' ya ve MQTT istemcisine teslim etmek için üç nitelikte hizmet sağlar: "en az bir kez", "en az bir kez" ve "tam olarak bir kez". Bir MQTT istemcisi, abonelik oluşturmak için WebSphere MQ ' ya bir istek gönderdiğinde, istek "en az bir kez" hizmet kalitesiyle gönderilir.

- Fark yok.

İleti kalıcılığı

“Message persistence in MQTT clients” sayfa 512	Publication iletileri, "en az bir kez" ya da "tam olarak bir kez" hizmet kalitesiyle gönderilirse, kalıcı kılınabilirler. İstemcide kendi kalıcılık mekanizmanızı uygulayabilir ya da istemciyle birlikte sağlanan varsayılan kalıcılık mekanizmasını kullanabilirsiniz. Kalıcılık, istemciye ya da istemciden gönderilen yayınlar için her iki yönde de çalışır.

- Dil bağlama farkları nedeniyle, C istemcisinden ileti sürekliliği mekanizmasını aşağıdaki gibi ayarlayın. Call the MQTT C client with one of the three options set as the fourth parameter to `MQTTClient_create`:

  `MQTTCLIENT_PERSISTENCE_DEFAULT`
  Bir dosya tabanlı kalıcılık, istemci platformuna özgü ayrıntılar.

  `MQTTCLIENT_PERSISTENCE_NONE`
  Veriler yalnızca bellede tutulur ve istemci durduğunda kaybedilir. Aygıtlara ilişkin WebSphere MQ Telemetry yardımcı programı yalnızca bu seçeneği destekler.

  `MQTTCLIENT_PERSISTENCE_USER`
  Kendi kalıcılık mekanizmanızı uygulamak için işlevler geliştirebilirsiniz. `MQTTClient_create` çağrısında, işlevlerinize işaret içeren `MQTTClient_persistence` bir yapıyı geçirin. Ayrıntılar için MQTT C istemci başvuru bilgilerini okuyun.

Temizleme oturumları

“Temizleme oturumları” sayfa 507	MQTT istemcisi ve telemetry (MQXR) hizmeti, oturum durumu bilgilerini korur. Devlet bilgileri, "en az bir kez" ve "tam olarak bir kez" teslimi sağlamak için ve "tam olarak bir kez" yayınların alıntısını sağlamak için kullanılır. Oturum durumu, bir MQTT istemcisi tarafından yaratılan abonelikleri de içerir. Oturumlar arasında durum bilgilerini korumadan ya da bakım yapmadan bir MQTT istemcisi çalıştırabilir. Bağlanmadan önce `MqttConnectOptions.cleanSession` ayarını yaparak temizleme oturumu kipini değiştirin.

- Fark yok.

Son irade ve vasiyet

“Son irade ve vasiyet yayını” sayfa 511	Bir MQTT istemci bağlantısı beklenmeyen bir şekilde sona ererse, WebSphere MQ Telemetry 'i "son bir irade ve ahit" yayını gönderecek şekilde yapılandırabilirsiniz. Yayının içeriğini ve gönderinin gönderileceği konuyu önceden tanımlayın. "Son irade ve vasiyet" bir bağlantı özellidir. İstemciyi bağlamadan önce bunu yaratın.

- Fark yok.

Program hatalarının işlenmesi

Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

Mümkün olduğunda, kuyruk yöneticisi bir MQI çağrısı yapıldığında hata döndürür. Bunlar, *yerel olarak belirlenen hatalardır*.

Uzak bir kuyruğa ileti gönderirken, MQI çağrısı yapıldığında hatalar görünmeyebilir. Bu durumda, hata raporlarını tanımlayan kuyruk yöneticisi, kaynak programa başka bir ileti göndererek bunları bildirir. Bunlar *uzaktan belirlenen hatalardır*.

Yerel olarak belirlenen hatalar

Yerel olarak saptanan hatalara ilişkin bilgiler: Bir MQI çağrısında hata, sistem kesintileri ve yanlış veri içeren iletiler.

Kuyruk yöneticisinin hemen rapor verebileceği en sık rastlanan üç hata nedeni:

- Bir MQI çağrısının başarısız olması; örneğin, kuyruğun dolu olduğu için
- Uygulamanızın bağlı olduğu sistemin bir kısmının çalıştırılmasına ilişkin bir kesinti; örneğin, kuyruk yöneticisi
- Başarıyla işlenemeyen verileri içeren iletiler

Zamanuyumsuz put olanağını kullanıyorsanız, hatalar hemen bildirilmez. Önceki zamanuyumsuz koyma işlemlerine ilişkin durum bilgilerini almak için MQSTAT çağrısını kullanın.

Bir MQI çağrısının başarısız olması

Kuyruk yöneticisi, bir MQI çağrısının kodlamasındaki hataları hemen bildirebilir. Bu, önceden tanımlanmış bir dönüş kodu kümesini kullanarak yapar. Bunlar, tamamlanma kodlarına ve neden kodlarına bölünüyorlar.

Bir çağrı başarılı olup olmadığını göstermek için, kuyruk yöneticisi arama tamamlandığında bir *tamamlama kodu* döndürür. Başarılı, kısmi tamamlama ve çağrı başarısızlığı belirten üç tamamlama kodu

vardır. Kuyruk yöneticisi ayrıca, kısmi tamamlama ya da çağrı hatasının nedenini belirten bir *neden kodu* değerini de döndürür.

Her bir çağrıya ilişkin tamamlanma ve neden kodları, Dönüş kodları' ta bu çağrıya ilişkin açıklamayla listelenir. Düzeltici eylemle ilgili fikirler de içinde olmak üzere daha ayrıntılı bilgi için bkz:

- Diğer tüm WebSphere MQ platformları içinNeden kodları

Her çağrıdan oluşabilecek tüm dönüş kodlarını işlemek için programlarınızı tasarlayın.

Sistem kesintileri

Bağlı olduğu kuyruk yöneticisinin bir sistem hatasından kurtulması durumunda, uygulamanız herhangi bir kesintiden habersiz olabilir. Ancak, bu tür bir kesinti oluştuğunda verilerinizin kaybedilmediğinden emin olmak için uygulamanızı tasarlamanız gerekir.

Verilerinizin tutarlı olduğundan emin olmak için kullanabileceğiniz yöntemler, kuyruk yöneticinizin çalıştığı altyapıya bağlıdır:

UNIX, Linux ve Windows sistemleri
Bu ortamlarda, MQPUT ve MQGET çağrılarınızı olağan biçimde yapabilirsiniz, ancak MQCMIT ve MQBACK çağrılarını kullanarak eşitleme noktalarını bildirmeniz gerekir (bkz. “İş birimlerinin kesinleştirilmesi ve yedeklenmesi” sayfa 308). CICS ortamında, MQCMIT ve MQBACK komutları, CICStarafından yönetilen iş birimleri içinde MQPUT ve MQGET çağrılarınızı yapabildiğiniz için devre dışı bırakılır.

Kaybetmeyi göze alamayadığınız tüm verileri taşımak için kalıcı iletiler kullanın. Kuyruk yöneticisi bir hatadan kurtarılması gerekiyorsa, kalıcı iletiler kuyruklara geri dönmektedir. With WebSphere MQ on UNIX, Linux, and Windows systems, an MQGET or MQPUT call within your application will fail at the point of filling all the log files, with the message MQRC_RESOURCE_PROBLEM. AIX, HP-UX, Linux, Solaris ve Windows sistemlerindeki günlük dosyalarıyla ilgili daha fazla bilgi için bkz. Yönetme.

Bir uygulama çalışırken kuyruk yöneticisi bir işletmen tarafından durdurulursa, susturma seçeneği genellikle kullanılır. Kuyruk yöneticisi, uygulamaların çalışmaya devam edebileceği bir susturma durumuna girer, ancak en kısa zamanda sona ermelidir. Küçük, hızlı uygulamalar, büyük olasılıkla susturulmuş durumu yoksayabilir ve normal olarak sona erdirilinceye kadar devam edebilir. Uygulamaların daha uzun çalıştırılması ya da iletilerin gelmesi beklenenler, MQOPEN, MQPUT, MQPUT1ve MQGET çağrılarını kullandıklarında *durdurulursa başarısız olur* seçeneğini kullanmalıdır. Bu seçenekler, kuyruk yöneticisi sustururken çağrıların başarısız olduğu anlamına gelir; ancak, uygulamanın susturulmuş durumdan yoksayma çağrıları yayınlayarak temizleme işlemini sonlandırma zamanı hala olabilir. Bu tür uygulamalar, gerçekleştirdikleri değişiklikleri de kesinleştirebilir ya da geri alabilir ve daha sonra sona erdirebilir.

Kuyruk yöneticisi durdurulmaya zorlandıysa (yani, susturulmuş durumdan dur), uygulamalar MQI çağrıları yaptıklarında MQRC_CONNECTION_BROKEN neden kodunu alır. Uygulamadan çıkın ya da diğer bir seçenek olarak UNIX, Linuxve Windows sistemlerinde bir MQDISC çağrısı yayınlayın.

Yanlış veri içeren iletiler

Uygulamandaki iş birimlerini kullandığınızda, bir program kuyruktan aldığı bir iletiyi başarılı bir şekilde işleyemezse, MQGET çağrısını geri alır.

Kuyruk yöneticisi, olan sayıların sayısının (ileti tanımlayıcısının *BackoutCount* alanında) sayısını tutar. Bu sayı, etkilenen her iletinin tanımlayıcısında bu sayıyı korur. Bu sayı, bir uygulamanın verimliliğiyle ilgili değerli bilgiler sağlayabilir. Zaman içinde artan geri sayım olan iletiler sürekli olarak reddedilir; uygulamanızı tasarlayın, böylece bu tür iletileri buna göre işler ve bu tür iletileri işler.

WebSphere MQ for Windows, UNIX ve Linux sistemlerinde, geriletme sayısı her zaman kuyruk yöneticisinin yeniden başlatılmasına neden olur.

Sorun belirleme için rapor iletilerini kullanma

Uzak kuyruk yöneticisi, bir MQI aramanızı yaparken bir iletiyi kuyruğa koyamamanız gibi hataları raporlayamaz, ancak iletiyi nasıl işlediğini bildiren bir rapor iletisi gönderebilir.

Uygulamanızın içinde, (MQPUT) rapor iletilerini ve bunları alma seçeneğini belirleyebilirsiniz (bu durumda başka bir uygulama ya da bir kuyruk yöneticisi tarafından gönderilirler).

Rapor iletileri oluşturma

Rapor iletileri, bir uygulamanın başka bir uygulamaya, gönderilen iletiyle baş edemeyeceğini bildirmesini sağlar.

Ancak, iletiyi gönderen uygulamanın herhangi bir sorun hakkında bilgilendirilmekle ilgilenip ilgilenmediğini belirlemek için *Report* alanının ilk olarak analiz edilmesi gerekir. Bir rapor iletisinin gerekli olduğunu belirlemiş olmak, aşağıdakine karar vermeniz gerekir:

- Özgün iletinin tamamını, yalnızca ilk 100 baytlık verileri içermek isteyip istemediğinizi ya da özgün iletinin hiçbirini içermeyeceğini.
- Özgün iletiyle ne yapılır? Bunu atabilir ya da ölü-mektup kuyruğuna gönderebilirsiniz.
- *MsgId* ve *CorrelId* alanlarının içindekilerin de gerekli olup olmadığını.

Oluşturulan rapor iletisinin nedenini belirtmek için *Feedback* alanını kullanın. Rapor iletilerinizi bir uygulamanın yanıtlama kuyruğuna yerleştirin. Ek bilgi için Geribildirim başlıklı konuya bakın.

İsteme ve alma (MQGET) rapor iletileri

Başka bir uygulamaya ileti gönderdiğinizde, gereksinim duyduğunuz geri bildirimi belirtmek üzere *Report* alanını tamamlamadığınız sürece herhangi bir sorun hakkında bilgilendirilmediğiniz anlamına gelir. Kullanılabilir seçenekler için bkz. Rapor alanı yapısı .

Kuyruk yöneticileri her zaman bir uygulamanın yanıtlama kuyruğuna rapor iletileri koyar ve kendi uygulamalarınızın aynı şekilde yapılması önerilir. Rapor iletisi olanağını kullandığınızda, iletinizin ileti tanımlayıcısında yanıtlanacak yanıtınızın adını belirtin; aksi takdirde, MQPUT çağrısı başarısız olur.

Uygulamanızın yanıt kuyruğunuzu izleyen yordamları içermesi ve ona gelen iletileri işlemeniz gerekir. Bir rapor iletisinin, özgün iletinin ilk 100 baytı, özgün iletinin ilk 100 baytı içerebileceğini ya da özgün iletinin hiçbirini içermediğini unutmayın.

Kuyruk yöneticisi, hata nedenini belirtmek için rapor iletisinin *Feedback* alanını ayarlar; örneğin, hedef kuyruk yok. Programlarınız aynı şeyi yapmalıdır.

Rapor iletilerine ilişkin daha fazla bilgi için bkz. “Rapor iletileri” sayfa 11.

Uzaktan saptanan hatalar

Uzak bir kuyruğa ileti gönderdiğinizde, yerel kuyruk yöneticisi bir hata bulmadan MQI aramanızı işlese bile, diğer etmenler iletinizin uzak bir kuyruk yöneticisi tarafından nasıl işleneceğini etkileyebilir.

Örneğin, hedeflediğiniz kuyruk dolu olabilir ya da kuyruk yok olabilir. İletiniz, hedef kuyruğa yönlendirilecek diğer ara kuyruk yöneticileri tarafından işlenecekse, bunların herhangi biri bir hata bulabilir.

İleti teslim edilmesi sorunları

Bir MQPUT çağrısı başarısız olduğunda, iletiyi yeniden kuyruğa yerleştirmeyi, gönderene geri döndürmeyi ya da bu iletiyi ölü harf kuyruğuna koymayı deneyebilirsiniz.

Her seçeneğin merits değeri vardır, ancak MQPUT ' un başarısız olmasının nedeni hedef kuyruğun dolu olduğu için bir ileti yeniden yerleştirmeyi denemek istemeyebilirsiniz. Bu örnekte, bunu ölü harf kuyruğuna koymak, bunu daha sonra doğru hedef kuyruğa ulaştırmanızı sağlar.

İleti sağlamayı yeniden dene

Before the message is put on a dead-letter queue, a remote queue manager attempts to put the message on the queue again if the attributes *MsgRetryCount* and *MsgRetryInterval* have been set for the channel, or if there is a retry exit program for it to use (the name of which is held in the channel attribute *MsgRetryExitId* field).

MsgRetryExitId alanı boş bırakılırsa, *MsgRetryCount* ve *MsgRetryInterval* özniteliklerindeki değerler kullanılır.

MsgRetryExitId alanı boş değilse, bu adın çıkış programı çalıştırılır. Kendi çıkış programlarınızı kullanma hakkında daha fazla bilgi için bkz. "İleti alışverişi kanallarına ilişkin kanal çıkışı programları" sayfa 378.

İletiyi gönderene geri ver

Özgün iletinin tümünü içermek üzere bir rapor iletisi oluşturulmasını isteyerek gönderene bir ileti dönmenizi sağlar.

Rapor iletisi seçeneklerine ilişkin ayrıntılar için bkz. "Rapor iletileri" sayfa 11 .

Ölü harf (teslim edilmemiş ileti) kuyruğunun kullanılması

Kuyruk yöneticisi bir iletiyi teslim edemediğinde, iletiyi ölüme mektup kuyruğuna yerleştirmeyi dener. Kuyruk yöneticisi kurulu olduğunda bu kuyruk tanımlanmalıdır.

Programlarınız, kuyruğun kullandığı kuyruğu, kuyruk yöneticisinin kullandığı aynı şekilde kullanabilir. Kuyruk yöneticisi nesnesini (MQOPEN çağrısını kullanarak) ve *DeadLetterQName* özniteliğini sorgulayarak (MQINQ çağrısını kullanarak), ölü-mektup kuyruğunun adını bulabilirsiniz.

Kuyruk yöneticisi bu kuyruğa bir ileti yerleştirdiğinde, iletiye üstbilgi ekler; bu biçim, ölü-mektup üstbilgisi (MQDLH) yapısı tarafından tanımlanır; bkz. MQDLH-Dead-letter header . Bu üstbilgi, hedef kuyruğun adını ve iletinin ölü harf kuyruğuna konmasının nedenini içerir. İleti, ileti istenen kuyruğa konmadan önce kaldırılmalı ve sorunun çözülmesi gerekir. Ayrıca, kuyruk yöneticisi, iletinin bir MQDLH yapısı içerdiğini göstermek için ileti tanımlayıcısının (MQMD) *Format* alanını değiştirir.

MQDLH yapısı

You are recommended to add an MQDLH structure to all messages that you put on the dead-letter queue; however, if you intend to use the dead-letter handler provided by certain WebSphere MQ products, you ***gerekir*** add an MQDLH structure to your messages.

İletinin üstbilgisinin eklenmesi, iletiyi ölü harf kuyruğu için çok uzun bir hale getirebileceğinden, iletilerinizin, en azından MQ_MSG_HEADER_LENGTH değişmezinin değeri tarafından, ölü-harfli kuyruk için izin verilen büyüklük üst sınırından daha kısa olmasına dikkat edin. Bir kuyruğa izin verilen ileti boyutu üst sınırı, kuyruğun *MaxMsgLength* özniteliğinin değerine göre belirlenir. Ölü-mektup kuyruğu için, bu özniteliğin kuyruk yöneticisi tarafından izin verilen üst sınıra ayarlandığından emin olun. Uygulamanız bir iletiyi teslim edemiyorsa ve ileti, ölü-mektup kuyruğuna konmak için çok uzun olursa, MQDLH yapısının tanımında verilen öneriyi izleyin.

Ölü harf kuyruğunun izlendiğinden ve bu kuyruğun üzerine gelen iletilerin işlendiğinden emin olun. Ölü-harfli kuyruk işleyicisi bir toplu iş yardımcı programı olarak çalışır ve seçilen iletiler üzerinde, seçilen iletiler kuyruğunda çeşitli işlemler gerçekleştirmek için kullanılabilir. Daha fazla ayrıntı için bkz. "Kuyruk-kuyruğun kuyruğunda işlenmesi" sayfa 530.

Veri dönüştürme gerekliyse, MQGET çağrısında MQGMO_CONVERT seçeneğini kullandığınızda, kuyruk yöneticisi üstbilgi bilgisini dönüştürür. İletiyi koyulan işlem bir MCA ise, üstbilgi, özgün iletinin tüm metni tarafından takip edilir.

Bu kuyruk için çok uzunsa, ölü-mektup kuyruğuna gönderilen iletiler kesilmiş olabilir. Bu durumun olası bir göstergesi, kuyruktaki iletiler, kuyruğun *MaxMsgLength* özniteliğinin değeriyle aynı uzunlukta olan iletilerdir.

Kuyruk-kuyruğun kuyruğunda işlenmesi

Bu bilgiler, kuyruk-harf kuyruğunda işlem yaparken genel kullanıma açık programlama arabirimi bilgilerini içerir.

Çıkmaz mektup kuyruğu işleme, yerel sistem gereksinimlerine bağlıdır, ancak belirtimi çizdiğinizde aşağıdaki şeyleri göz önünde bulundurun:

- MQMD ' deki biçim alanının değeri MQFMT_DEAD_LETTER_HEADER olduğu için, ileti, ölü-mektup kuyruğu üstbilgisine sahip olduğu tanımlanabilir.
- On WebSphere MQ for z/OS using CICS, if an MCA puts this message to the dead-letter queue, the *PutApplType* field is MQAT_CICS, and the *PutApplName* field is the *ApplId* of the CICS system followed by the transaction name of the MCA.
- İletinin ölü-mektup kuyruğuna yöneltilmesinin nedeni, ölü-harfli kuyruk üstbilgisinin *Reason* alanında yer alır.
- Ölü-harfli kuyruk üstbilgisi, hedef kuyruk adı ve kuyruk yöneticisi adına ilişkin ayrıntıları içerir.
- Ölü-mektup kuyruğu üstbilgisi, ileti hedef kuyruğuna konmadan önce ileti tanımlayıcısında yeniden yürürlüğe konması gereken alanları içerir. Bu bilgiler şunlardır:
  1. *Encoding*
  2. *CodedCharSetId*
  3. *Format*
- İleti tanımlayıcısı, gösterilen üç alan (Encoding, CodedCharSetIdve Format) dışında, özgün uygulama tarafından PUT ile aynıdır.

Ölü harf kuyruğu uygulamanızın aşağıdaki şeylerden birini ya da birkaçını yapması gerekir:

- *Reason* alanını inceleyin. Aşağıdaki nedenlerle bir ileti MCA tarafından konulmış olabilir:
  - İleti, kanala ilişkin ileti boyutu üst sınırından daha uzun

    Neden: MQRC_MSG_TOO_BIG_FOR_CHANNEL
  - İleti hedef kuyruğuna konamadı.

    Neden, bir MQPUT işlemi tarafından döndürülebilen herhangi bir MQRC_ * neden koddur.
  - Bir kullanıcı çıkışı bu işlemi istedi

    Neden kodu kullanıcı çıkışıyla ya da varsayılan MQRC_SUPPRESSED_BY_EXIT tarafından sağlanır.
- İletiyi amaçlanan hedefe iletmeye çalışın, bu mümkün olabilir.
- Şaşırtma nedenini saptmadan önce belirli bir süre boyunca iletiyi alıkoyma, ancak hemen düzeltilebilir değil.
- Yöneticilere, bunların belirlendiği yerlerde doğru sorunları çözmeleri için yönergeler verin.
- Bozuk ya da uygulanamaz olan iletileri atın.

Ölü harf kuyruğundan kurtardığınız iletilerle başa çıkmak için iki yol vardır:

1. İleti yerel bir kuyruksa:
   - Uygulama verilerini çıkarmak için gereken kod çevirilerini gerçekleştirme
   - Bu yerel bir işlevse, bu verilerde kod dönüştürmeleri gerçekleştirmeyi sağlar
   - Sonuç iletisini yerel kuyruğa, geri yüklenen ileti tanımlayıcısının tüm ayrıntılarıyla birlikte yerel kuyruğa koyun
2. İleti uzak bir kuyruk için gönderilmişse, iletiyi kuyruğa koyun.

Teslim edilmeyen iletilerin dağıtılmış bir kuyruğa alma ortamında nasıl işlendiği hakkında bilgi için bkz. Bir ileti teslim edilemezse ne olur?.

Çoklu yayın programlama

Bir kuyruk yöneticisine bağlanma ve kural dışı durum raporlaması gibi WebSphere MQ Multicast programlama görevleri hakkında bilgi edinmek için bu bilgileri kullanın.

WebSphere MQ Multicast, mümkün olduğu kadar kullanıcı için şeffaf olacak şekilde tasarlandı ve yine de var olan uygulamalarla uyumlu olacak şekilde tasarlandı. Bir COMMINFO nesnesi tanımlama ve KONU nesnesinin **MCAST** ve **COMMINFO** parametrelerinin ayarlanması, var olan WebSphere MQ uygulamalarının çoklu yayını kullanmak için önemli bir yeniden yazma gerektirmediği anlamına gelir. Ancak, bazı sınırlamalar olabilir (ek bilgi için bkz. “Çoklu Yayın ve İleti Kuyruğu Arabirimi” sayfa 531) ve dikkat edilmesi gereken bazı güvenlik sorunları (ek bilgi için bkz. Çoklu Yayın Güvenliği).

Çoklu Yayın ve İleti Kuyruğu Arabirimi

Ana MQI kavramlarını ve bunların WebSphere MQ Multicast ile nasıl ilişkilendirileceğini anlamak için bu bilgileri kullanın.

Çok hedefli abonelikler kalıcı değildir; ilgili fiziksel kuyruklar olmadığından, kalıcı abonelikler tarafından oluşturulan çevrimdışı iletilerin saklanacak bir yeri yoktur.

Bir uygulama çok noktaya gönderim konusuna abone olduktan sonra, bir nesne tanıtıcısı gibi, kuyruğun kullanabileceği bir nesne tanıtıcısı ya da MQGET işlemi tarafından verilir. Bu, yalnızca yönetilen birden çok noktaya gönderim aboneliklerinin (MQSO_YANED ile yaratılan abonelikler) desteklendiği anlamına gelir; bu, abonelik yapmak ve iletileri bir kuyrukta 'noktası' olarak göstermek olanaklı değildir. Bunun anlamı, iletilerin abonelik çağrısında döndürüldüğü nesne tanıtıcısından tüketilmesi gerektiği anlamına gelir. İstemcide iletiler, istemci tarafından tüketilinceye kadar bir ileti arabelleğiyle saklanır; ek bilgi için bkz. İstemci yapılandırma dosyasınınMessageBuffer kısmı . İstemci yayınlama hızına uymuyorsa, iletiler gerektiği gibi atılır ve en eski iletiler önce atılır.

Genellikle, bir uygulamanın KONU nesnesinin MCAST özniteliği ayarlanarak, uygulamanın Multicast ya da not Multicast özelliğini kullanıp kullanmadığı bir yönetim karardır. Bir yayınlama uygulamasının çoklu yayın kullanımının kullanılmamasını sağlaması gerekiyorsa, bu uygulama `MQOO_NO_MULTICAST` seçeneğini kullanabilir. Similarly, a subscribing application can ensure that multicast is not used by subscribing with the `MQSO_NO_MULTICAST` option.

WebSphere MQ Multicast ileti seçicilerinin kullanımını destekler. Bir uygulama, bir uygulama tarafından yalnızca, seçim dizgisinin temsil ettiği SQL92 sorgusuna uyan özelliklere sahip olan iletilere kaydolmak için kullanılır. İleti seçiciyle ilgili daha fazla bilgi için bkz. “Seçiciler” sayfa 21.

Aşağıdaki çizelge, tüm ana MQI kavramlarını ve bunların Multicast ile nasıl ilişkilendirileceğini göstermektedir:

Çizelge 71. MQI kavramları ve bunların çok hedefli olarak nasıl ilişkilendirildikleri

MQI Kavramı	Çoklu yayın kullanılarak denendiğinde işlem	Neden Kodu
Sıfır uzunluklu bir ileti koymak	Reddedildi	2005 (07D5) (RC2005): MQRC_BUFFER_LENGTH_ERROR
Gruplandırma	Reddedildi	2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR
Bölümleme	Reddedildi	2443 (098B) (RC2443): MQRC_SEGMENTATION_NOT_ALLOWEND
Dağıtım listeleri	Reddedildi	2154 (086A) (RC2154): MQRC_RECS_PRESENT_ERROR

Çizelge 71. MQI kavramları ve bunların çok hedefli olarak nasıl ilişkilendirildikleri (devamı var)

MQI Kavramı	Çoklu yayın kullanılarak denendiğinde işlem	Neden Kodu
MQINQ	Konular için reddedilen konular: MQINQ ve MQSET konuları desteklenmiyor.	2038 (07F6) (RC2038): MQRC_NOT_OPEN_FOR_SORGULAMA
	Yönetilen tanıtıcı için kabul edildi. Yalnızca `Yürürlükteki Derinlik` sorgulanabilir.	• Değer Yürürlükteki Derinlik ise, geçerli bir neden kodu yoktur. • Değer, Yürürlükteki Derinlik değerinden başka bir değerse, neden kodu 2067 (0813) (RC2067): MQRC_SELECTOR_ERROR.
MQSET	Tüm tutamaçlar için reddedildi.	2040 (07F8) (RC2040): MQRC_NOT_OPEN_FOR_SET
İşlemler (XA ya da değil)	Reddedildi	2072 (0818) (RC2072): MQRC_SYNCPOINT_NOT_AVALABILIR
İleti göz atma	Reddedildi	2036 (07F4) (RC2036): MQRC_NOT_OPEN_FOR_BROWSE
İletileri kilitle	Reddedildi	2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR
İşaretle göz at	Reddedildi	2036 (07F4) (RC2036): MQRC_NOT_OPEN_FOR_BROWSE
Geçiş bağlamı	Reddedildi	2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR
MQPUT1	Reddedildi. Bir Multicast only konusuna denemek için MQPUT1 ' yi geçersiz kılır.	2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY
Sürekli Abonelik	Konu "Yalnızca çoklu yayın" olarak işaretlenmişse, tersi durumda, Multicast aboneliği olmayan bir abonelik yapılır.	2436 (0984) (RC2436): MQRC_DURABILITY_NOT_ALLOWALIZE
TopicString > 255	Reddedildi. Konu dizesi 255 karakterden fazlaysa, istemcide reddedilir.	2425 (0979) (RC2425): MQRC_TOPIC_STRING_ERROR

Çizelge 71. MQI kavramları ve bunların çok hedefli olarak nasıl ilişkilendirildikleri (devamı var)		
MQI Kavramı	**Çoklu yayın kullanılarak denendiğinde işlem**	**Neden Kodu**
Yönetilmeyen abonelik yapıldı	Konu "Yalnızca çoklu yayın" olarak işaretlenmişse, tersi durumda, Multicast aboneliği olmayan bir abonelik yapılır.	2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR
MQPMO_NOT_OWN_SUBS	Reddedildi	2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR

Aşağıdaki öğeler, önceki çizelgeden bazı MQI kavramlarını genişletebilir ve çizelgede olmayan bazı MQI kavramlarına ilişkin bilgi sağlar:

İleti kalıcılığı
Kalıcı olmayan çok noktaya yayın aboneleri için, yayınlayıcıdan gelen kalıcı iletiler kurtarılamaz bir şekilde teslim edilir.

İleti kesme
İletinin kesilmesi destekleniyor; bu da, uygulamanın aşağıdaki gibi bir uygulama için mümkün olduğunu gösterir:

1. Bir MQGET komutu verin.
2. MQRC_TRUNCATED_MSG_BAŞARISIZ oldu.
3. Daha büyük bir arabellek ayırın.
4. İletiyi almak için MQGET ' yi yeniden yayınlayın.

Abonelik süre bitimi
Abonelik süre bitimi desteklenmiyor. Süre bitimi ayarlama girişimi yoksayılır.

Çok noktaya gönderim için yüksek kullanılabilirlik

WebSphere MQ Multicast kesintisiz eşdüzeyler arası işlemi anlamak için bu bilgileri kullanın; ancak WebSphere MQ bir WebSphere MQ kuyruk yöneticisine bağlansa da, iletiler kuyruk yöneticiliklerinden geçmez.

Bir kuyruk yöneticisine yönelik bir bağlantının MQOPEN ya da MQSUB için çok hedefli konu nesnesi için yapılması gerekse de, iletiler kuyruk yöneticisinde kendilerini aklamamalıdır. Bu nedenle, çok hedefli konu nesnesinde MQOPED ya da MQSUB tamamlandıktan sonra, kuyruk yöneticisine yönelik bağlantı kaybolsa bile çoklu yayın iletilerini iletme işlemine devam etmek mümkündür. İki işlem kipi vardır:

Kuyruk yöneticisine olağan bir bağlantı yapılır
Kuyruk yöneticiyle bağlantı varsa, çoklu yayın iletişimi mümkün. Bağlantı başarısız olursa, normal MQI kuralları uygulanır; örneğin, çok hedefli nesne tanıtıcısında bir MQPUT işlemi 2009 (07D9) (RC2009): MQRC_CONNECTION_BROKENdeğerini döndürür.

Kuyruk yöneticisinden bir istemci bağlantısı yeniden bağlanıyor
Yeniden bağlantı döngüsü sırasında bile çoklu yayın iletişimi mümkün. Başka bir deyişle, kuyruk yöneticisine yönelik bağlantı kesilse bile, çoklu yayın iletilerinin yerleştirilmesinin ve tüketilmesinin etkilenmemesi anlamına gelir. İstemci bir kuyruk yöneticisine yeniden bağlanmayı dener ve bu yeniden bağlantı başarısız olursa, bağlantı tanıtıcısı bozuk olur ve çoklu yayın olanlar da içinde olmak üzere tüm MQI çağrıları başarısız olur. Ek bilgi için şu konuya bakın: Automated client reconnection

Herhangi bir uygulama bir MQDISC ' yi belirtik olarak yayınlarsa, tüm çoklu yayın abonelikleri ve nesne tanıtıcıları kapatılır.

Çok hedefli sürekli eşdüzeyler arası işlem

İstemciler arasındaki eşdüzeyler arası iletişimin avantajlarından biri, iletilerin kuyruk yöneticisi aracılığıyla akmasına gerek kalmaması; dolayısıyla, kuyruk yöneticisi ile bağlantı kesmesi durumunda, ileti aktarımı devam eder. Bu kipin sürekli ileti gereksinimleri için aşağıdaki kısıtlamalar geçerlidir:

- Kesintisiz işlem için MQCNO_RECONNECT_ * seçeneklerinden biri kullanılarak bağlantı yapılmalıdır. Bu işlem, iletişim oturumu bozuk olsa da, gerçek bağlantı tanıtıcısı bozulmamış olsa da, bunun yerine yeniden bağlanma durumunda olduğu anlamına gelir. Yeniden bağlanma başarısız olursa, bağlantı tanıtıcısı artık bozuk olur ve bu da tüm MQI çağrılarını önler.
- Bu kipte yalnızca MQPUT, MQGET, MQINQ ve Async Consume desteklenmektedir. Herhangi bir MQOP, MQCLOSE ya da MQDISC fiillerinin tamamlanması için kuyruk yöneticisine yeniden bağlantı yapılması gerekir.
- Durum, kuyruk yöneticisi durağına akar; kuyruk yöneticisinde herhangi bir durum eski ya da eksik olan herhangi bir durum olabilir. Bu, istemcilerin iletileri gönderip alabileceği ve kuyruk yöneticisininde herhangi bir durumun bilinmediği anlamına gelir. Daha fazla bilgi için bakınız: Multicast application monitoring

Çok hedefli ileti alışverişi için MQI ' de veri dönüştürme

WebSphere MQ Multicast ileti sistemi için veri dönüştürme çalışmalarının nasıl çalıştığını anlamak için bu bilgileri kullanın.

WebSphere MQ Multicast paylaşılan, bağlantısız bir iletişim kuralıdır ve bu nedenle her istemcinin veri dönüştürmesi için belirli istekler hazırlamaması mümkün değildir. Aynı çok noktaya yayın akışına abone olan her istemci aynı ikili verileri alır; bu nedenle, WebSphere MQ veri dönüştürme gerekiyorsa, dönüştürme her istemcide yerel olarak gerçekleştirilir.

Data is converted on the client for WebSphere MQ Multicast traffic. **MQGMO_CONVERT** seçeneği belirtilirse, veri dönüştürme işlemi istendiği gibi yapılır. Kullanıcı tanımlı biçimler, istemcide kurulu veri dönüştürme çıkışa gereksinim duyarlar; istemci ve sunucu paketlerinde şimdi kitaplıkların hangi kitaplıkların olduğunu görmek için bkz. “Veri dönüştürme çıkışları yazılıyor” sayfa 397 .

Veri dönüştürmenin yönetilmesine ilişkin bilgi için Multicast ileti alışverişi için veri dönüştürmenin etkinleştirilmesibaşlıklı konuya bakın.

Veri dönüştürme hakkında daha fazla bilgi için bkz. Veri dönüştürme.

Veri dönüştürme çıkışları ve `ClientExitPath`ile ilgili daha fazla bilgi için, istemci yapılandırma dosyasının ClientExitYol kısmına bakın.

Çok noktaya yayın kural dışı durumu

WebSphere MQ Multicast olay işleyicileri ve raporlama WebSphere MQ Multicast kural dışı durumları hakkında bilgi edinmek için bu bilgileri kullanın.

WebSphere MQ Multicast, standart WebSphere MQ olay işleyici mekanizması kullanılarak raporlanan çok noktaya yayın olaylarını raporlamak için olay işleyicisini çağırarak sorun belirlemeye yardımcı olur.

Tek bir çok noktaya gönderim olayı, birden çok WebSphere MQ olayının çağrılması nedeniyle sonuçlanabilir; bunun nedeni, aynı çoklu yayın vericisini ya da alıcısını kullanan birden çok `MQHCONN` bağlantı tanıtıcısı olabilir. Ancak, her çok noktaya yayın kural dışı durumu, WebSphere MQ bağlantısı başına tek bir olay işleyicinin çağrılmasına neden olur.

WebSphere MQ `MQCBDO_EVENT_CALL` sabiti, uygulamaların yalnızca WebSphere MQ olaylarını alacak bir geri bildirme kaydetmesini sağlar ve `MQCBDO_MC_EVENT_CALL` , uygulamaların yalnızca çok hedefli olayları alacak bir geri bildirme kaydetmesini sağlar. Her iki değişmez de kullanılırsa, her iki olay tipi de alınır.

Multicast olayları istenmesi

WebSphere MQ Multicast olayları, `cbd.Options` alanında `MQCBDO_MC_EVENT_CALL` değişmezini kullanır. Aşağıdaki örnek, çok hedefli olayların nasıl isteneceğini göstermektedir:

```
cbd.CallbackType     = MQCBT_EVENT_HANDLER;
cbd.Options          = MQCBDO_MC_EVENT_CALL;
cbd.CallbackFunction = EventHandler;
MQCB(Hcon,MQOP_REGISTER,&cbd,MQHO_UNUSABLE_HOBJ,NULL,NULL,&CompCode,&Reason);
```

`cbd.Options` alanı için `MQCBDO_MC_EVENT_CALL` seçeneği belirtildiğinde, olay işleyici, bağlantı düzeyi olayları yerine yalnızca WebSphere MQ Multicast olaylarını göndermektedir. To request that both types of events are sent to the event handler, the application must specify the `MQCBDO_EVENT_CALL` constant in the `cbd.Options` field as well as the `MQCBDO_MC_EVENT_CALL` constant as shown in the following example:

```
cbd.CallbackType     = MQCBT_EVENT_HANDLER;
cbd.Options          = MQCBDO_EVENT_CALL | MQCBDO_MC_EVENT_CALL
cbd.CallbackFunction = EventHandler;
MQCB(Hcon,MQOP_REGISTER,&cbd,MQHO_UNUSABLE_HOBJ,NULL,NULL,&CompCode,&Reason);
```

Bu değişmezlerden hiçbiri kullanılmazsa, olay işleyiciye yalnızca bağlantı düzeyi olayları gönderilir.

`Options` alanına ilişkin değerler hakkında daha fazla bilgi için bkz. Seçenekler (MQlong).

Çok hedefli olay biçimi

WebSphere MQ Multicast kural dışı durumları, geri çağırma işlevinin **`Buffer`** değiştirgesinde döndürülen destekleyici bazı bilgileri içerir. **`Buffer`** işaretçisi bir işaretçiler dizisini işaret eder ve `MQCBC.DataLength` alanı, dizinin bayt cinsinden büyüklüğünü belirtir. Dizinin ilk öğesi her zaman olayın kısa bir metin açıklamasına işaret eder. Olayın tipine bağlı olarak daha fazla parametre sağlanabilir. Aşağıdaki çizelge kural dışı durumları listeler:

Çizelge 72. Çok hedefli olay kodu açıklamaları

Olay Kodu	Tanım	Ek veriler
MQMCEV_PACKET_LOSS	Kurtarılamayan paket kaybı	Kaybedilen paket sayısı
MQMCEV_HEARTBEAT_TIMEOUT	Sağlıklı işletim bildirimi paketinin uzun olmaması	Uyglnmz
MQMCEV_VERSION_CONFLIP	Daha yeni protokol sürümü paketleri alma	Uyglnmz
MQMCEV_GÜVENILIRLIĞI	Verici ve alıcı vericinin farklı güvenilirlik kipleri	Uyglnmz
MQMCEV_CLOSED_TRANS	Konu iletimi 1 kaynak tarafından kapatılmış	Uyglnmz
MQMCEV_STREAM_ERROR	Akışta hata saptandı	Uyglnmz
MQMCEV_NEW_SOURCE	Konu üzerinde yeni bir kaynak iletilmeye başlar	Kaynak yapı
MQMCEV_RECEIVE_QUEUE_TRIMLED	Zaman ya da alan süre bitimi nedeniyle PacketQ ' dan kaldırılan paketler	Kırpılan paketlerin sayısı
MQMCEV_PACKET_LOSS_NACK_EXPLORY	NACK süre bitimi nedeniyle kurtarılamayan paket kaybı	Kaybedilen paket sayısı
MQMCEV_ACK_RETRIES_EXCEED	**`max_ack_retries`** aşıldıktan sonra geçmişten kaldırılan paketler	Kaldırılan paket sayısı

Çizelge 72. Çok hedefli olay kodu açıklamaları (devamı var)

Olay Kodu	Tanım	Ek veriler
MQMCEV_STREAM_SUSPEND_NACK	Bu konu tarafından kabul edilen bir akışta NACK ' lar askıya alındı	Akış tanıtıcısını askıya al
		Akımın askıya alındığı süre (milisaniye olarak)
MQMCEV_STREAM_RESUME_NACK	NACK ' lar bir akışta askıya alındıktan sonra sürdürülüyor	Akış Tanıtıcısı
MQMCEV_STREAM_ATILDI	Bu konu tarafından kabul edilen bir akış, bir except isteği nedeniyle reddedildi	Akış Tanıtıcısı
MQMCEV_FIRST_ILETISI	Bir kaynaktan ilk ileti	İleti numarası
MQMCEV_LATE_JOIN_FAILURE	Geciken birleştirme oturumu başlatılamadı	Uyglnmz
MQMCEV_MESSAGE_LOSS	Kurtarılamaz ileti kaybı	Kaybedilen iletilerin sayısı
MQMCEV_SEND_PACKET_FAILURE	Çok noktaya yayın vericisi çok noktalı bir paket gönderemedi	Uyglnmz
MQMCEV_REPAIR_DELAY	Çok hedefli alıcı, bekleyen bir NAK için onarım paketi almadı	Uyglnmz
MQMCEV_MEMORY_ALERT_ON	Alıcı alma arabellekleri dolduruyor	Arabellek havuzu kullanım yüzdesi
MQMCEV_MEMORY_ALERT_OFF	Alıcı alma arabellekleri olağan durumda.	Arabellek havuzu kullanım yüzdesi
MQMCEV_NACK_ALERT_ON	Alıcı onarımı paket isteği hızı, yüksek su işaretine ulaştı	Saniye başına paketlerde geçerli onarım isteği hızı
MQMCEV_NACK_ALERT_OFF	Günlük nesnesi onarma paketi istek hızı aşağı doğru	Saniye başına paketlerde geçerli onarım isteği hızı
MQMCEV_REPAIR_ALERT_ON	Verici onarımı paket gönderme hızı, yüksek su işaretine ulaştı	Uyglnmz
MQMCEV_REPAIR_ALERT_OFF	Verici onarma paketi gönderme hızı normale indi	Uyglnmz
MQMCEV_SHM_DEST_EWISIFY	Bir verici konu hedefi tarafından kullanılan Paylaşılan Bellek bölgesinin kullanılamaz olduğu saptandı.	Uyglnmz
MQMCEV_SHM_PORT_KULLANILAMAZ	Bir günlük nesnesi eşgörünümü tarafından kullanılan Paylaşılan Bellek kapısının kullanılamadığı saptandı.	Uyglnmz
MQMCEV_CCT_GETTIME_FAILED	Eşgüdümlü küme zamanından alma süresi başarısız oldu	Uyglnmz
MQMCEV_DEST_INTERFACE_FAILURE	Bir verici konu hedefi tarafından kullanılan ağ arabirimi başarısız oldu ve bir yedek ağ arabirimi kullanılamıyor	

Çizelge 72. Çok hedefli olay kodu açıklamaları (devamı var)		
Olay Kodu	**Tanım**	**Ek veriler**
MQMCEV_DEST_INTERFACE_FAILOVER	Bir verici konu hedefi tarafından kullanılan ağ arabirimi başarısız oldu ve başka bir Arabirim için başarılı bir hata durumunda yedek sisteme geçiş işlemi tamamlandı.	
MQMCEV_PORT_INTERFACE-HATA	rmmPort nesnesi tarafından kullanılan ağ arabirimi başarısız oldu ve bir yedek ağ arabirimi kullanılamıyor (ya da başarısız da var)	RMM yapılandırması
MQMCEV_PORT_INTERFACE_FAILOVER	The network interface used by a receiver rmmPort has failed and a successful failover to another Interface has been completed	RMM yapılandırması

.NET kullanımı

.NET için WebSphere MQ sınıfları, .NET programlama çerçevesinde yazılmış bir programın WebSphere MQ ' a bir WebSphere MQ MQI istemcisi olarak bağlanmasını ya da doğrudan bir WebSphere MQ Server sunucusuna bağlanmasını sağlar.

Microsoft .NET Framework kullanan uygulamalariniz varsa ve WebSphere MQolanaklarından yararlanmak istiyorsanız, .NET için WebSphere MQ sınıflarını kullanmanız gerekir.

Nesne yönelimli WebSphere MQ .NET arabirimi, MQI fiillerini kullanmak yerine nesne yöntemlerini kullandığı MQI arabiriminden farklıdır.

Yordamsal WebSphere MQ uygulama programlama arabirimi, aşağıdaki listede yer alan fiiller etrafında oluşturulmuştur:

```
MQCONN, MQDISC, MQOPEN, MQCLOSE,
MQINQ, MQSET, MQGET, MQPUT, MQSUB
```

Bu fiillerin tümü, bir parametre olarak, çalışacakları WebSphere MQ nesnesi için bir tanıtıcı olarak alır. .NET nesne yönelimli olduğu için .NET programlama arabirimi bu turu dönüştürür. Programınız, bu nesnelerle ilgili yöntemler çağırarak işlem yapmak istediğiniz bir WebSphere MQ nesnelerinden oluşur. Programları .NET tarafından desteklenen herhangi bir dilde yazabilirsiniz.

Yordamsal arabirimi kullanırken, MQDISC çağrısını (*Hconn*, CompCode, Reason) kullanarak bir kuyruk yöneticisinden bağlantınız kesilir; burada *Hconn* , kuyruk yöneticisine ilişkin bir tanıtıcıdır.

.NET arabiriminde, kuyruk yöneticisi, MQQueueManagersınıfı bir nesle temsil edilir. Bu sınıftaki Disconnect () yöntemini çağırarak kuyruk yöneticisinden bağlantıyı kesmenizi sağlar.

```
// declare an object of type queue manager
MQQueueManager queueManager=new MQQueueManager();
...
// do something...
...
// disconnect from the queue manager
queueManager.Disconnect();
```

WebSphere MQ classicas for .NET, .NET uygulamalarının WebSphere MQile etkileşimde bulunabilmesini sağlayan bir sınıf kümesidir. Bunlar, uygulamanızın kullandığı (kuyruk yöneticileri, kuyruklar, kanallar ve iletiler gibi) WebSphere MQ ' un çeşitli bileşenlerini temsil eder. Bu sınıfların ayrıntıları için WebSphere MQ .NET sınıflarına ve arabirimlerinebakın.

Yazdığınız herhangi bir uygulamayı derleyebilmek için önce .NET Framework kurulu olmalıdır. .NET ve .NET Framework için WebSphere MQ sınıflarının kurulmasına ilişkin yönergeler için bkz. “.NET için WebSphere MQ sınıflarının kurulması” sayfa 539.

İlgili kavramlar

Teknik genel bakış

“Bağlanma seçenekleri” sayfa 538
Bir kuyruk yöneticisine .NET için WebSphere MQ sınıflarının bağlanmasını içeren üç kip vardır. Gereksinimlerinize en uygun bağlantı tipini göz önünde bulundurun.

“.NET için WebSphere MQ sınıflarının kullanılması” sayfa 549
Bu konu grubunda, .NET kuruluşu için WebSphere MQ sınıflarınızı ve kendi programlarınızı nasıl çalıştıracağını doğrulamak üzere örnek programları çalıştırmak için sisteminizi nasıl yapılandıracağını ele alan konular açıklanmaktadır.

“WebSphere MQ .NET problemlerinin çözülmesi” sayfa 551
Bir program başarıyla tamamlanmazsa, örnek uygulamalarından birini çalıştırın ve tanılama iletilerinde verilen önerileri izleyin.

“ WebSphere MQ .NET programlarının yazılması ve konuşlandırılması” sayfa 552
.NET için WebSphere MQ sınıflarını WebSphere MQ kuyruklarına erişmek üzere kullanmak için, programları, iletileri içeren çağrılar içeren .NET tarafından desteklenen herhangi bir dilde yazar ve WebSphere MQ kuyruklarından ileti alır.

“Microsoft Windows Communication Foundation (WCF) içinIBM WebSphere MQ özel kanalı” sayfa 571
The Microsoft Windows Communication Foundation (WCF) custom channel for IBM WebSphere MQ sends and receives messages between WCF clients and services.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“Uygulamaların geliştirilmesi” sayfa 7
IBM WebSphere MQ , iş süreçlerinizi desteklemek için gereksinim duyardığınız iletileri göndermek ve almak için uygulamalar geliştirebileceğiniz çeşitli yollar sağlar. Kuyruk yöneticilerinizi ve ilgili kaynaklarınızı yönetmek için de uygulamalar geliştirebilirsiniz.

.NET için WebSphere MQ sınıflarıyla çalışmaya başlama

.NET için WebSphere MQ sınıfları, .NET programlama çerçevesinde yazılmış bir programın WebSphere MQ ' a bir WebSphere MQ MQI istemcisi olarak bağlanmasını ya da doğrudan bir WebSphere MQ Server sunucusuna bağlanmasını sağlar.

Bağlanma seçenekleri

Bir kuyruk yöneticisine .NET için WebSphere MQ sınıflarının bağlanmasını içeren üç kip vardır. Gereksinimlerinize en uygun bağlantı tipini göz önünde bulundurun.

İstemci bağ tanımları bağlantısı

To use WebSphere MQ classes for .NET as a WebSphere MQ MQI client, you can install it, with the WebSphere MQ MQI client, either on the WebSphere MQ server machine, or on a separate machine. İstemci bağ tanımları bağlantısı XA ya da XA dışı hareketleri kullanabilir

Sunucu bağ tanımları bağlantısı

Sunucu bağ tanımları kipinde kullanıldığında, .NET için WebSphere MQ class for .NET, bir ağ üzerinden iletişim kurmak yerine kuyruk yöneticisi API 'sini kullanır. This provides better performance for WebSphere MQ applications than using network connections.

To use the bindings connection, you must install WebSphere MQ classes for .NET on the WebSphere MQ server.

Yönetilen istemci bağlantısı

Bu kipte yapılan bir bağlantı, yerel ya da uzak bir makinede çalışan bir WebSphere MQ istemcisi olarak WebSphere MQ istemcisi olarak bağlanır.

Bu kipteki .NET bağlantısı için WebSphere MQ sınıfları .NET yönetilen kodunda kalır ve yerel hizmetlere çağrılar yapmamaz. Yönetilen kodla ilgili ek bilgi için Microsoft belgelerine bakın.

Yönetilen istemciyi kullanmak için bir dizi sınırlama vardır. Bunlarla ilgili daha fazla bilgi için bkz. "Yönetilen istemci bağlantıları" sayfa 552.

.NET için WebSphere MQ sınıflarının kurulması

Örnekler de içinde olmak üzere, .NET için WebSphere MQ sınıfları WebSphere MQile birlikte kurulur. Microsoft .NET Framework için bir önkoşul vardır.

The latest version of WebSphere MQ classes for .NET is installed by default as part of the standard WebSphere MQ installation in the *Java ve .NET Messaging ve Web Hizmetleri* feature. Kuruluş yönergeleri için bkz. Installing IBM WebSphere MQ server on Windows ya da Installing an IBM WebSphere MQ client on Windows systems .

In a multiple installation environment, if you have previously installed the WebSphere MQ classes for .NET as a support pack, you cannot install WebSphere MQ unless you first uninstall the support pack. WebSphere MQ ile kurulan .NET özelliğine ilişkin WebSphere MQ sınıfları, destek paketiyle aynı işlevselliği içerir.

Kaynak dosyalar da içinde olmak üzere örnek uygulamalar da sağlanır; bkz. "Örnek Uygulamalar" sayfa 549.

32 bit ya da 64 bit altyapılarında .NET için WebSphere MQ sınıflarını çalıştırmak için Microsoft .NET Framework V2.0 ya da üstünü kurmuş olmanız gerekir.

Not: WebSphere MQ V7.0.1kuruluşundan önce Microsoft .NET Framework v2.0 ya da üstü kurulu değilse, WebSphere MQ ürün kuruluşu hatasız olarak devam eder, ancak .NET için WebSphere MQ sınıfları kullanılamayacaktır. If the .NET Framework is installed after installing WebSphere MQ 7.0.1, then the WebSphere .NET assemblies must be registered by running the *WMQInstallDir*`\bin\amqiRegisterdotNet.cmd` script, where *WMQInstallDir* is the directory where WebSphere MQ 7.0.1 is installed. Bu komut dosyası, gerekli düzenekleri Global Assembly Cache (GAC) içine kurar. Alınan işlemleri kaydeden bir `amqi*.log` dosyası kümesi, %TEMP% dizininde oluşturulur.

.NET 3 ile Microsoft WCF için WebSphere MQ özel kanalının kullanılmasıyla ilgili bilgi edinmek için bkz: "Microsoft Windows Communication Foundation (WCF) içinIBM WebSphere MQ özel kanalı" sayfa 571

.NET ' te dağıtımlı hareketler

Dağıtılmış hareketler ya da genel hareketler, istemci uygulamalarının bir işlemdeki iki ya da daha çok ağa bağlı sisteme çeşitli veri kaynaklarını içermesine olanak sağlar.

Dağıtımlı işlemlerde, bir hareket yöneticisi iki ya da daha fazla kaynak yöneticisi arasındaki hareketi düzenler ve yönetir.

Hareketler tek faz ya da iki aşamalı kesinleştirme işlemi olabilir. Tek aşamalı kesinleştirme, hareket ve iki aşamalı kesinleştirme işleminde yalnızca bir kaynak yöneticisinin katılacağı bir işlemdir ve harekette yer alan birden çok kaynak yöneticisi vardır. İki aşamalı kesinleştirme işleminde, hareket yöneticisi, tüm kaynak yöneticilerinin kesinleştirilmeye hazırlanıp hazırlanmadığını denetlemek için bir hazırlama çağrısı gönderir. Tüm kaynak yöneticilerinden alındı bildirimi alındığında, kesinleştirme çağrısı yayınlanır. Tersi durumda, işlemin tamamında geriye işleme gerçekleşir. Daha fazla ayrıntı için bkz. Hareket yönetimi ve destek . Kaynak yöneticileri, harekete katılımlarının işlem yöneticilerini bilgilendirmelidir. Kaynak yöneticisi, katılımının hareket yöneticisine bilgi veriyorsa, hareket kesinleştirme ya da geri alma işlemi olduğunda, kaynak yöneticisi hareket yöneticisinden geri çağrılar alır.

WebSphere MQ .NET sınıfları, yönetilmeyen ve sunucu bağ tanımları kipi bağlantılarında dağıtılmış hareketleri önceden destekler. Bu kiplerde, WebSphere MQ .NET sınıfları, tüm çağrılarını, .NET adına hareket işlemesini yöneten C genişletilmiş hareket istemcisine aktarır.

WebSphere MQ.NET sınıfları artık, dağıtılmış hareketler desteği için WebSphere MQ .NET Sınıflarının System.Transactions ad alanını kullandığı yönetilen kipte dağıtılmış hareketleri destekler. System.Transactions altyapısı, WebSphere MQdahil olmak üzere tüm kaynak yöneticilerinde başlatılan işlemleri destekleyerek işlemsel programlama basit ve verimli hale getirir. WebSphere MQ .NET uygulaması, .NET örtük hareket programlama ya da belirtik hareket programlama modeli kullanarak ileti alabilir ya da alabilir. Örtük işlemlerde, hareket sınırları, ne zaman kesinleştirileceğine, geriye işlenmeye (belirtik işlemler için) karar veren uygulama programı tarafından yaratılır ya da işlemi tamamlar. Belirtik işlemlerde, kesinleştirmek, geri almak ve işlemi tamamlamak isteyip istemediğinizi belirtik olarak belirtmeniz gerekir.

WebSphere MQ.NET , hareket yöneticisi olarak Microsoft dağıtılmış hareket eşgüdümcüsü (MS DTC) kullanır ve birden çok kaynak yöneticisi arasında işlemi düzenler ve yönetir. WebSphere MQ , kaynak yöneticisi olarak kullanılır.

WebSphere MQ.NET , X/Open Distributed Transaction Processing (DTP) modelinden sonra gelir. X/Open Distributed Transaction Processing modeli, bir satıcı konsorsiyumu olan Open Group (Açık Grup) tarafından önerilen dağıtılmış bir hareket işleme modesidir. Bu model, işlem işleme ve veritabanı etki alanlarında ticari satıcı firmaların çoğu arasında bir standarttır. Ticari işlem yönetimi ürünlerinin çoğu, X/DTP modelini destekler.

Hareket kipleri

- “Yönetilen kipteki dağıtılmış hareketler” sayfa 541
- Yönetilmeyen kip için dağıtılmış hareketler

Çeşitli senaryolarda işlemlerin koordine edilmesi

- Bir bağlantı birkaç harekette katılabilir, ancak herhangi bir zamanda yalnızca bir işlem etkindir.
- İşlem sırasında, MQQueueManager.Bağlantı kesme çağrısı onurlandırılır. Bu durumda, işlemin geri işlenmesi istenir.
- Bir işlem sırasında, MQQueue.Close ya da MQTopic.Close çağrısı kabul edilir. Bu durumda, işlemin geri istenmesi istenir.
- Hareket sınırları, işlemin ne zaman kesinleştirileceğine, geri alınmasına (belirtik işlemler için) ya da işlemin tamamlanmasını (örtük işlemler için) karar veren uygulama programı tarafından yaratılır.
- Bir kuyruğa ya da konu çağrısına çağrı göndermeden ya da çağrı göndermeden önce istemci uygulaması beklenmeyen bir hatayla başarısız olursa, hareket geriye işlenir ve bir MQException yayınlanır.
- MQCC_FAILED neden kodu bir Kuyruk ya da Konu çağrısı sırasında döndürülürse ya da çağrıya çağrılırsa, neden kodu ile bir MQException yayınlanır ve hareket işlenir. Bir hazırlama çağrısı hareket yöneticisi tarafından önceden verildiyse, WebSphere MQ .NET işlemi, işlemi zorla geri döndürerek hazırlama isteğini döndürür. Daha sonra, hareket yöneticisi DTC, geçerli ortam işlemlerindeki tüm kaynak yöneticileriyle birlikte yürürlükteki çalışma için geriye işleme neden olur.
- Birden çok kaynak yöneticisi içeren bir işlem sırasında, bazı ortam nedeni Koma ya da Alma çağrısının süresiz olarak askıda kalma sürmesine neden olursa, hareket yöneticisi öngörülme süreye kadar bekler. Bu süre geçtikten sonra, yürürlükteki ortam hareketlerindeki tüm kaynak yöneticileriyle tüm geçerli çalışmaların geriye işlenmesine neden olur. Hazırlık aşamasında bu süresiz bekleme gerçekleşirse, hareket yöneticisi zamanaşımına uğrayabilir ya da işlemin geriye işlendiği durumlarda, kaynak üzerinde belirsiz bir çağrı yayınlayabilir.
- İşlemleri kullanan uygulamaların SYNC_POINT altına ileti koyması ya da iletileri almaları gerekir. Bir ileti koyma ya da alma çağrısı, SYNC_POINT altında olmayan bir işlemsel bağlam altında yayınlanırsa, çağrıya MQRC_UNIT_OF_WORK_NOT_STARTED neden koduyla başarısız olur.

Microsoft .NET System.Transactions ad alanı kullanılarak Yönetilen ve Yönetilmeyen İstemci işlem desteği arasındaki davranış farkları

Nested Transactions have a TransactionScope inside another TransactionScope

- WebSphere MQ .NET fully Managed Client, iç içe geçmiş TransactionScope' u destekler
- WebSphere MQ .NET yönetilmeyen istemci iç içe geçmiş TransactionScopeöğesini desteklemez

System.Transactions' dan Bağımlı İşlemler

- WebSphere MQ .NET tam olarak yönetilen istemci, System.Transactionstarafından sağlanan bağımlı hareketler olanağını destekler.
- WebSphere MQ .NET yönetilmeyen istemci, System.Transactionstarafından sağlanan bağımlı hareket olanağını desteklemez.

Ürün Örnekleri

Yeni ürün örnekleri `SimpleXAPut`ve `SimpleXAGet` , `WebSphere MQ\tools\dotnet\samples\cs\base` altında kullanılabilir. Örnekler, SystemTransactions ad alanını kullanarak Dağıtılmış Hareketler altında MQPUT ve MQGET kullanılmasını gösteren C# uygulamalarıdır. Bu örneklerle ilgili daha fazla bilgi için bkz. “Creating simple put and get messages within a TransactionScope” sayfa 544.

Yönetilen kipteki dağıtılmış hareketler

WebSphere MQ .NET sınıflarında, yönetilen kipteki dağıtılmış işlemler desteği için System.Transactions ad alanı kullanılır. Yönetilen kipte, MS DTC, bir harekette listelenen tüm sunucularda dağıtılmış hareketleri koordine eder ve yönetir.

WebSphere MQ .NET classes provide an explicit programming model based on the System.Transactions.Transaction class and an implicit programming model using the System.Transactions.TransactionScope, class where the transactions are automatically managed by the infrastructure.

Örtük İşlem

Aşağıdaki kod parçası, bir WebSphere MQ .NET uygulamasının .NET örtük hareket programlamasını kullanarak bir iletiyi nasıl yerleştirdiğini açıklamalı.

```
Using (TransactionScope scope = new TransactionScope ())
{
    Q.Put (putMsg,pmo);
    scope.Complete ();
}

Q.close();
qMgr.Disconect();}
```

Örtük hareketin kod akışına ilişkin açıklama

Kod, *TransactionScope* öğesini yaratır ve iletiyi kapsam altına yerleştirir. Daha sonra, işlemin tamamlanmasına ilişkin işlem koordinatörü bilgilendirmek için *Complete* (Tamamlandı) çağrısını çağırır. Artık hareket eşgüdümcüsü hareketi tamamlamak için *prepare* ve *commit* konularını yayınlar. Bir sorun saptanırsa, bir *geri alma* çağrılır.

Belirtik İşlem

Aşağıdaki kod, bir WebSphere MQ .NET uygulamasının, .NET açık hareket programlama modelini kullanarak iletileri nasıl yerleştirdiğini açıklamalı.

```
MQQueueManager qMgr = new MQQueuemanager ("MQQM);
MQQueue Q = QMGR.AccessQueue("Q", MQC.MQOO_OUTPUT+MQC.MQOO_INPUT_SHARED);
MQPutMessageOptions pmo = new MQPutMessageOptions();
pmo.Options = MQC.MQPMO_SYNCPOINT;
MQMessage putMsg1 = new MQMessage();
Using(CommittableTransaction tx = new CommittableTransaction()){
Transaction.Current = tx;
    try
    {
    Q.Put(MSG,pmo);
    tx.commit();
    }
    catch(Exception)
    {tx.rollback();}
    }
```

```
Q.close();
qMgr.Disconnect();
}
```

Belirtik hareket kod akışının açıklaması

Kod parçası, *CommitableTransaction* sınıfını kullanarak hareket yaratır. Bu, o kapsam altına bir ileti koyar ve daha sonra, hareketi tamamlamak için belirtik olarak *commit* (kesinleştirme) çağrılarını çağırır. Herhangi bir sorun varsa *geriye işleme* çağrılır.

Yönetilmeyen kipteki dağıtılmış hareketler

WebSphere MQ.NET sınıfları, genişletilmiş hareket istemcisi ve COM + /MTS kullanan yönetilmeyen bağlantıları (istemci), örtük ya da belirtik hareket programlama modelini kullanarak hareket koordinatörü olarak destekler. Yönetilmeyen kipte, WebSphere MQ .NET sınıflarında, tüm çağrıları, .NET adına hareket işlemesini yöneten C genişletilmiş hareket istemcisine devrir.

Hareket işleme, bir dış hareket yöneticisi tarafından denetlenir ve hareket yöneticisinin API 'si denetimi altında genel iş birimi eşgüdümünün eşgüdümlerini sağlar. MQBEGIN, MQCMIT ve MQBACK fiilleri kullanılamıyor. WebSphere MQ .NET sınıfları, bu desteği, yönetilmeyen taşıma kipi (C istemcisi) yoluyla gösterir. Bkz. XA uyumlu hareket yöneticilerinin yapılandırılması

MTS, CICS, Tuxedo ve diğer platformlarda mevcut olduğu şekilde Windows NT ' ta aynı özellikleri sağlamak için bir hareket işleme (TP) sistemi olarak gelişmiştir. MTS kurulduğunda, Microsoft Distributed Transaction Coordinator (MSDTC) olarak adlandırılan Windows NT ' e ayrı bir hizmet eklenir. MSDTC, ayrı veri depolarına ya da kaynaklarına yayılan işlemleri koordine eder. Çalışmak için, her veri deposunun kendi özel kaynak yöneticisini gerçekleştirmesi gerekir.

WebSphere MQ , DTC XA çağrılarını WebSphere MQ(X/Open) çağrılarına eşlemeyi yönettiği bir arabirim (özel kaynak yöneticisi arabirimi) uygulayarak MSDTC ile uyumlu hale gelir. WebSphere MQ , bir kaynak yöneticisinin rolünü yürütür.

COM + gibi bir bileşen bir WebSphere MQ' ya eriştiğinde, COM genellikle uygun MTS bağlam nesnesiyle denetleyerek bir işlem yapılması gerektiğini denetler. Bir işlem gerekiyorsa, COM DTC ' yi bilgilendirir ve bu işlem için otomatik olarak integral WebSphere MQ işlemini başlatır. Daha sonra, COM, MQMTS yazılımıyla veri ile birlikte çalışır, iletileri doldurur ve gerektiği gibi iletiler elde eder. COM ' tan elde edilen nesne eşgörünümü, verilerdeki tüm işlemler sona erdikten sonra SetComplete ya da SetAbort yönteminden geçmektedir. When the application issues SetComplete, the call signals the DTC that the application has completed the transaction and the DTC can go ahead with the two-phase commit process. The DTC then issues calls to MQMTS which in turn issues calls to WebSphere MQ to commit or roll back the transaction.

Yönetilmeyen istemciyi kullanarak bir WebSphere MQ .NET uygulaması yazılması

COM + bağlamı içinde çalıştırmak için, .NET sınıfı System 'den edinmelidir.EnterpriseServices.ServicedComponent. Hizmet verilen bileşenleri kullanan düzenekler oluşturmak için kurallar ve öneriler aşağıdaki gibi olur:

Not: Aşağıdaki adımlar yalnızca System.EnterpriseServices kipini kullanıyorsanız anlamdır.

- COM + içinde başlatılmakta olan sınıf ve yöntem için genel (iç sınıf yok, korunan ya da statik yöntemler yok) olmalıdır.
- Sınıf ve yöntem öznitelikleri: TransactionOption özniteliği, sınıfın hareket düzeyini belirtir; bu öznitelik, işlemlerin devre dışı bırakılıp bırakılmadığını, desteklendiğini ya da gerekli olup olmadığını gösterir. ExecuteUOW() yöntemindeki AutoComplete özniteliği, işlenmeyen bir kural dışı durum yayınlamazsa, COM + ' a hareketi kesinleştirmesini bildirir.
- Düzeneğin güçlü bir şekilde adlandırılması: Düzeneğin, Global Assembly Cache (GAC) içinde güçlü bir şekilde adlandırılması ve kayıtlı olması gerekir. Düzenek, GAC ' de kaydedildikten sonra açık olarak ya da tembel kayıt tarafından COM + içinde kayıtlı.
- COM + içinde bir düzeneğin kaydedilmesi: Düzeneğin COM istemcilerine açıklanabilmesini hazırlayın. Then create a type library by using the Assembly Registration tool, regasm.exe.

```
regasm UnmanagedToManagedXa.dll
```

- Düzeneği GAC `gacutil /i UnmanagedToManagedXa.dll`içine kaydedin.
- .NET services kuruluş programı aracını (regsvcs.exe) kullanarak, birleştirmeyi COM + içinde kaydedin. regasm.exe:tarafından yaratılan tip kitaplığına bakın.

```
Regsvcs /appname:UnmanagedToManagedXa  /tlb:UnmanagedToManagedXa.tlb
UnmanagedToManagedXa.dll
```

- Düzenek GAC 'ye konuşlandırılır ve daha sonra tembel kayıt tarafından COM +' da kayıtlı olur. .NET çerçevesi, koddan sonra ilk kez çalıştırıldıktan sonra kayıt bakımını üstalır.

COM + ile System.EnterpriseServices modelini ve System.Transactions işlevini kullanarak örnek kod akışı aşağıdaki kısımlarda açıklanmıştır:

System.EnterpriseServices modelini kullanan örnek kod akışı

```
using System;
using IBM.WMQ;
using IBM.WMQ.Nmqi;
using System.Transactions;
using System.EnterpriseServices;

namespace UnmanagedToManagedXa
{

[ComVisible(true)]
[System.EnterpriseServices.Transaction(System.EnterpriseServices.TransactionOption.Required)
]
    public class MyXa : System.EnterpriseServices.ServicedComponent
    {

        public MQQueueManager QMGR = null;
        public MQQueueManager QMGR1 = null;
        public MQQueue QUEUE = null;
        public MQQueue QUEUE1 = null;
        public MQPutMessageOptions pmo = null;
        public MQMessage MSG = null;

        public MyXa()
        {
        }

        [System.EnterpriseServices.AutoComplete()]
        public void ExecuteUOW()
        {
            QMGR = new MQQueueManager("usemq");

            QUEUE = QMGR.AccessQueue("SYSTEM.DEFAULT.LOCAL.QUEUE",
                                     MQC.MQOO_INPUT_SHARED +
                                     MQC.MQOO_OUTPUT +
                                     MQC.MQOO_BROWSE);

            pmo = new MQPutMessageOptions();
            pmo.Options = MQC.MQPMO_SYNCPOINT;
            MSG = new MQMessage();
            QUEUE.Put(MSG, pmo);
            QMGR.Disconnect();
        }
    }
}


public void RunNow()
{
   MyXa xa = new MyXa();
   xa.ExecuteUOW();
}
```

COM + ile etkileşimler için System.Transactions komutunu kullanarak kod akışı örneği

```
[STAThread]
public void ExecuteUOW()
{
Hashtable t1 = new Hashtable();
t1.Add(MQC.CHANNEL_PROPERTY, "SYSTEM.DEF.SVRCONN");
t1.Add(MQC.HOST_NAME_PROPERTY, "localhost");
t1.Add(MQC.PORT_PROPERTY, 1414);
t1.Add(MQC.TRANSPORT_PROPERTY, MQC.TRANSPORT_MQSERIES_CLIENT);
```

```
TransactionOptions opts = new TransactionOptions();

using(TransactionScope scope = new  TransactionScope(TransactionScopeOption.RequiresNew,
                                opts,     EnterpriseServicesInteropOption.Full)
  {

        QMGR = new MQQueueManager("usemq", t1);
        QUEUE = QMGR.AccessQueue("SYSTEM.DEFAULT.LOCAL.QUEUE",
                                        MQC.MQOO_INPUT_SHARED +
                                        MQC.MQOO_OUTPUT +
                                        MQC.MQOO_BROWSE);

         pmo = new MQPutMessageOptions();
         pmo.Options = MQC.MQPMO_SYNCPOINT;
         MSG = new MQMessage();
         QUEUE.Put(MSG, pmo);
         scope.Complete();
  }
 QMGR.Disconnect();
}
```

Creating simple put and get messages within a TransactionScope

Ürün örnek C# uygulamaları WebSphere MQiçinde bulunur. Bu basit uygulamalar, bir TransactionScopeiçinde ileti koymanın ve iletilerin yerleştirilmesini gösterir. Görevin sonunda, bir kuyruktan ya da konudan iletiler yerleştirebilir ve iletiler alabilirsiniz.

Başlamadan önce

MSDTC hizmeti, XA İşlemleri için çalışıyor ve etkinleştirilmelidir.

Bu görev hakkında

Örnek, yalın bir uygulamadır (`SimpleXAPut` ve `SimpleXAGet`). `SimpleXAPut` ve `SimpleXAGet` programları, WebSphere MQ' da bulunan C# uygulamalarıdır. `SimpleXAPut` demonstrates using MQPUT, under Distributed Transactions using SystemTransactions namespace. `SimpleXAGet` demonstrates using MQGET, under Distributed Transactions using SystemTransactions namespace.

`SimpleXAPut` , `WebSphere MQ\tools\dotnet\samples\cs\base`içinde bulunur

Yordam

Uygulamalar, `tools\dotnet\samples\cs\base\bin`komutundan komut satırı değiştirgeleriyle çalıştırılabilir.

```
SimpleXAPut.exe -d destinationURI [-h host -p port -l channel -tx transaction -tm mode -n
numberOfMsgs]
```

```
SimpleXAGet.exe -d destinationURI [-h host -p port -l channel -tx transaction -tm mode -n
numberOfMsgs]
```

parametrelerin bulunduğu yer:

`-destinationURI`

Bu, kuyruk ya da konu olabilir. Bir kuyruk için, `queue://queueName` olarak ve bir konunun `topic://topicName`olarak belirtilmesini belirtin.

`-host`

Bu, localhost ya da bir IP adresi gibi bir anasistem adı olabilir.

`-port`

Kuyruk yöneticisinin çalışmakta olduğu kapı.

`-channel`

Kullanılmakta olan bağlantı kanalı. Varsayılan değer `SYSTEM.DEF.SVRCONN`

-transaction

Hareket sonucu; örneğin, kesinleştirme ya da geriye işleme gibi.

-mode

Örneğin, yönetilen ya da yönetilmeyen taşıma kipi.

-numberOfMsgs

İletilerin sayısı. Varsayılan değer 1'dir.

Örnek

```
SimpleXAPut -d topic://T01 -h localhost -p 2345 -tx rollback -tm unmanaged
```

```
SimpleXAGet -d queue://Q01 -h localhost -p 2345 -tx rollback -tm unmanaged
```

İşlemler Kurtarılıyor

Bu kısımda, yönetilen kip kullanılarak WebSphere MQ .NET XA içindeki işlemlerin kurtarılabilmesinin işlenmesi açıklanmaktadır.

Genel Bakış

Dağıtımlı hareket işleme işlemlerinde işlemler başarıyla tamamlanabilir. Ancak, bir işlemin birçok nedenden dolayı başarısız olabileceği senaryolar da olabilir. Bu nedenler arasında bir sistem arızası, donanım hatası, ağ hatası, yanlış ya da geçersiz veriler, uygulama hataları ya da doğal ya da insan yapımı olağanüstü durumlar bulunabilir. İşlem başarısızlıklarının önlenmesi olanaklı değildir. Dağıtılmış hareket sisteminin bu hataları işleme yeteneğine sahip olması gerekir. Hata ortaya çıktığında hataları saptayabilir ve düzeltebilmelidir. Bu işlem, İşlem Kurtarma olarak bilinir.

Dağıtımlı hareket işleme işleminin önemli bir yönü, eksik ya da belirsiz hareketlerin kurtarılması. Kurtarılması, belirli bir işlemin İş Birimi bölümü kurtarılıncaya kadar kilitli tutulmak üzere çalıştırılmalıdır. System.Transactions sınıf kitaplığından Microsoft .NET, tamamlanmamış/belirsiz hareketlerin kurtarılmasına ilişkin seçeneği sağlar. This recovery support expects Resource Manager to maintain the transaction logs and run the recovery when in need.

Kurtarma Modeli

Microsoft .NET ' in işlem kurtarma modelinde, Transaction Manager (System.Transactionsya da Microsoft Distributed Transaction Coordinator (MS DTC) ya da her ikisi), başlatılır, koordinatlar ve hareket kurtarma işlemini denetler. OLE Tx İletişim Kuralı (Microsoft XA protokolü) tabanlı Kaynak Yöneticileri, DTC ' yi sürücü, koordinat ve bunların kurtarılması için yapılandırma seçeneklerini sağlar. Bunu yapmak için, Kaynak Yöneticileri 'nin XA_Switch 'i MS DTC ile yerel arabirim kullanılarak kaydettirmesi gerekir.

XA_Switch provides the entry points of XA functions like xa_start, xa_end, and xa_recover in the Resource Manager to the Distributed Transaction Coordinator.

Microsoft Distributed Transaction Coordinator (DTC) kullanılarak kurtarma:

Microsoft Distributed Transaction eşgüdümcüsü, iki tür kurtarma işlemi sağlar.

Soğuk Kurtarma

Bir XA kaynak yöneticisine bağlantı açıksa, hareket yöneticisi işlemi başarısız olursa, soğuk kurtarma gerçekleştirilir. Hareket yöneticisi yeniden başlatıldığında, hareket yöneticisi günlüklerini okur ve XA kaynak yöneticiyle bağlantıyı yeniden kurar ve kurtarma işlemini başlatır.

Sıcak Kurtarma

XA kaynak yöneticisi ya da ağ başarısız olduğu için, hareket yöneticisi ile XA kaynak yöneticisi arasındaki bağlantı başarısız olursa, hareket yöneticisi devam ederse, çalışırken kurtarma işlemi gerçekleştirilir. Başarısızlığın ardından, hareket yöneticisi belirli aralıklarla XA kaynak yöneticisine

yeniden bağlanmayı dener. Bağlantı yeniden kurulduğunda, hareket yöneticisi XA kurtarma işlemini başlatır.

System.Transactions ad alanı, hareket yöneticisi olarak MS DTC ' ye dayalı olarak yönetilen dağıtımlı hareketlerin yönetilen somutlamasını sağlar. MS DTC ' nin yerel arabirimiyle aynı özellikleri sağlar, ancak tam olarak yönetilen ortamdır. Tek fark, işlem kurtarma işlemleriyle ilgilidir. System.Transactions , Kaynak Yöneticilerinin kurtarmayı kendi kendilerine kullanmasını ve daha sonra, İşlem Yöneticileri (MS DTC) ile koordinat etmesini bekler. Resource Manager , belirli bir tamamlanmamış işlemin kurtarılması için sormalı ve İşlem Yöneticisi bu işlemin gerçek sonucuna dayalı olarak onu ve koordinatları kabul eder.

WebSphere MQ .NET için işlem kurtarma işlemi

Bu bölümde, dağıtılmış hareketlerin WebSphere MQ .NET sınıflarıyla nasıl kurtarılabildiği ele alınmıştır.

Genel Bakış

Tamamlanmamış bir hareketi kurtarmak için, kurtarma bilgilerinin gerekli olduğunu kabul edin. İşlem kurtarma bilgileri, kaynak yöneticileri tarafından depolamak üzere günlüğe kaydedilmelidir. WebSphere MQ .NET sınıfları, benzer bir yolu izler. İşlem kurtarma bilgileri, SYSTEM.DOTNET.XARECOVERY.QUEUE.

WebSphere MQ .NET ' teki işlem kurtarma işlemi, iki aşamalı bir işlemdir.

1. İşlem kurtarma bilgilerinin günlüğe kaydedilmesi.
   - Hazırlama aşaması sırasında, her işlem için, kurtarma bilgilerini içeren kalıcı bir ileti SYSTEM.DOTNET.XARECOVERY.QUEUE.
   - Kesinleştirme çağrısı başarılı olursa, ileti silinir.
2. Bir Monitor uygulaması WmqDotnetXAMonitor kullanılarak işlem kurtarılıyor.
   - WmqDotnetXAMonitor, SYSTEM.DOTNET.XARECOVERY.QUEUE ve tamamlanmamış hareketleri kurtarır

   MCA iletiyi hedef kuyruğa koyamıyorsa, özgün iletiyi içeren bir kural dışı durum raporu oluşturur ve bunu, özgün iletide belirtilen yanıtlama kuyruğuna gönderilecek bir iletim kuyruğuna koyar. (Yanıtlama kuyruğu, MCA ile aynı kuyruk yöneticisiyse, ileti bir iletim kuyruğuna değil, doğrudan o kuyruğa konadır.)

SYSTEM.DOTNET.XARECOVERY.QUEUE

Bu, tamamlanmamış hareketlere ilişkin işlem kurtarma bilgilerini içeren bir sistem kuyruğdur. Bu kuyruk, bir kuyruk yöneticisi yaratıldığında yaratılır.

Not: SYSTEM.DOTNET.XARECOVERY.QUEUE KUYRı

WMQDotnetXAMonitor Uygulaması

WebSphere MQ .NET XA Monitor uygulaması, verili bir kuyruk yöneticisini izler ve varsa, eksik hareketleri kurtarır. Aşağıda eksik işlem olarak kabul edilir ve kurtarılır:

Tamamlanmamış Hareketler

- Hareket hazırlandıysa, ancak zamanaşımı süresi içinde COMMIT işlemi tamamlanmadıysa.
- Hareket hazırlandıysa, ancak WebSphere MQ Kuyruk yöneticisi sona ermiş olabilir.
- İşlem hazırlandıysa, ancak İşlem Yöneticisi sona erdiyse.

İzleme uygulaması, WebSphere MQ .NET istemci uygulamanızın çalıştığı aynı sistemden çalıştırılmalıdır. Aynı kuyruk yöneticisine bağlanan birden çok sistemde çalışan uygulamalar varsa, izleme uygulaması tüm sistemlerden çalıştırılmalıdır. Her istemci makinenin, uygulamayı kurtarmak için çalışan bir izleme uygulaması varsa, her izleme programı, yürürlükteki izleme programının yerel MS DTC ' nin yeniden listeleyebilmesi ve tamamlanabilmesi için eşgüdümleme yaptığı işleme karşılık gelen iletiyi belirleyebilmelidir.

WebSphere MQ .NET ' e ilişkin işlem kurtarma kullanımı

Farklı kullanım senaryoları aşağıda yer alıyor:

- **WebSphere MQ Tek DTC ve tek kuyruk yöneticisi yönetim ortamı kullanan uygulama:** Bu senaryoda, hareket altında kuyruk yöneticisine ve çalışan İş Birimini (UoW) bağlarken ve işlem başarısız olursa ve tamamlanamazsa, izleme uygulaması hareketi kurtarır ve işlemi tamamlar.

  Bu senaryoda, tek bir kuyruk yöneticisi işlemlerle ilişkili olduğu için, izleme uygulamasının tek bir eşgörünümü çalışır durumda olacaktır.
- **Tek DTC ve tek kuyruk yöneticisi yönetim ortamını kullanan birden çok WebSphere MQ uygulaması:** Bu senaryoda, tek DTC altında birden çok WMQ uygulaması vardır ve bunların tümü aynı kuyruk yöneticisine bağlanıyor ve hareketler altında UoW çalıştırılıyor.

  Hareketler başarısız olursa ve tamamlanamazsa, izleme uygulaması bunları kurtarır ve tüm uygulamalarla ilgili işlemleri tamamlar.

  Bu senaryoda, hareketlerde bir kuyruk yöneticisi kullanıldığı için tek bir izleme programı çalıştırılır.
- **Birden çok WebSphere MQ Applications, birden çok DTCs, farklı Kuyruk Yöneticisi örnekleri:** Bu senaryoda, birden çok WMQ uygulaması farklı DTM ' ler altında (yani, her uygulama farklı bir makinede çalıştırılıyor) ve farklı kuyruk yöneticilerine bağlanıyor.

  Hata oluşursa ve işlem tamamlanamazsa, Monitor uygulaması, DTC adresini saptamak için iletide yer alan TransactionManager' ın neresinde olduğunu denetler. TransactionManager, izleme programının çalışmakta olduğu DTC adresiyle eşleşiyorsa, kurtarma işlemini tamamlar; başka bir işlem, DTC ' ye karşılık gelen ileti bulununcaya kadar aramaya devam eder.

  Bu senaryoda, her istemcinin işlemlerde kullandığı kendi kuyruk yöneticisine sahip olduğu için, her istemci için (kullanıcı ya da bilgisayar) çalışan bir Monitor uygulamasının yalnızca bir eşgörünümü olacaktır.
- **Birden çok WebSphere MQ Applications, birden çok DTCs, birden çok aynı Kuyruk Yöneticisi örneği:** Bu senaryoda, farklı DTM ' ler altında birden çok WMQ uygulaması var (her uygulama farklı bir makinede çalıştırılıyor) ve tümü aynı kuyruk yöneticisine bağlanıyor.

  If failure occurs and transaction becomes incomplete, monitor application verifies the TransactionManagerWhereabouts in the message to check if the DTC address and value match with the DTC under which the monitor is running. Her iki değer de eşleşirse, kurtarma işlemi, DTC ' ye karşılık gelen iletiyi buluncaya kadar aramayı sürdürmeye devam eder.

  Bu senaryoda, her istemcinin hareketlerde kullandığı kendi kuyruk yöneticisi ilişkilendirmesi olduğu için, her istemci için (kullanıcı ya da bilgisayar) çalışan tek bir Monitor uygulaması eşgörünümü olacaktır.
- **Birden çok WebSphere MQ Applications, tek DTC, farklı Kuyruk Yöneticisi örnekleri:** Bu senaryoda, tek bir DTC altında birden çok WMQ uygulaması (bir bilgisayarda, çalışan birden çok WMQ uygulaması var) ve farklı kuyruk yöneticilerine bağlanıyor.

  Hareket başarısız olursa ve tamamlanamazsa, Monitor Application işlemi kurtarır.

  Bu senaryoda, her uygulamanın hareketlerde kullanılan kendi kuyruk yöneticisi olduğu ve her birinin kurtarılması gerektiği için, kuyruk yöneticisi olarak çalışan izleme uygulamasının birden çok eşgörünümü olacaktır.

  Not: İzleme uygulaması arka planda çalışmıyorsa, uygulamayı başlatabilirsiniz.

WMQDotnetXAMonitor uygulamasının kullanılması

XA izleyicisi uygulaması el ile çalıştırılmalıdır. Her an başlatılabilir. You can start it when you see the messages on the SYSTEM.DOTNET.XARECOVERY.QUEUE or you can keep it running in the background before you do any transactional work with the applications that are written using WebSphere MQ .NET classes.

Monitor uygulamasını başlatmak için kullanılan komut

```
WmqDotnetXAMonitor.exe -m <QueueManagerName> -n <ConnectionName> -c <Channel> -i
```

Nerede

- **n** -Anasistem (kapı) biçiminde bağlantı adı. Bağlantı adı birden çok bağlantı adı içerebilir. Virgülle ayrılmış listelerde birden çok bağlantı adı verilmelidir; örneğin, "localhost (1414), localhost (1415), localhost (1416)". Monitor uygulaması, virgülle ayrılmış listede belirtilen bağlantı adlarının her biri için kurtarma işlemini çalıştırır.
- **c** -Kanal adı.
- **m** -Kuyruk yöneticisi adı. İsteğe Bağlı
- **i** -Heuristic dalının tamamlanması. İsteğe Bağlı

İzleyici uygulaması aşağıdaki işlemleri gerçekleştirir:

1. SYSTEM.DOTNET.XARECOVERY.QUEUE (KUYRUK) 100 saniyenin bir aralığında.
2. Kuyruk derinliği sıfırdan büyükse, XA monitörü ileti kuyruğunu tarar ve iletinin tamamlanmamış işlem ölçütlerine uygun olup olmadığını denetler.
3. İletilerin herhangi biri tamamlanmamış işlem ölçütlerine uyursa, monitörü dışarı çeker ve işlem kurtarma bilgilerini alır.
4. Daha sonra, kurtarma bilgilerinin yerel MS DTC ile ilgili olup olmadığını belirler. If evet ise, işlemi geri almak için devam eder. Tersi durumda, sonraki iletiye göz atmak için geri döner.
5. Daha sonra, eksik işlemi kurtarmak için kuyruk yöneticisine çağrı yapar.

WmqDotNETXAMonitor uygulama yapılanışı dosyası ayarları
Uygulamayı izlemek için, uygulama yapılandırma dosyası kullanılarak girişler de sağlanabilir. Örnek bir uygulama yapılandırma dosyası WebSphere MQ .NET ile birlikte gönderilir. Bu dosya gereksinimlerinize göre değiştirilebilir.

Uygulama Yapılandırması dosyası, giriş değerlerini dikkate alırken en yüksek önceliği alır. Giriş değerleri hem komut satırı, hem de Uygulama Yapılandırması dosyasında sağlandıysa, uygulama yapılandırmasındaki değerler dikkate alınır.

Örnek uygulama yapılandırma dosyası.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
<configSections>
<sectionGroup name="IBM.WMQ">
<section name="dnetxa" type="System.Configuration.NameValueFileSectionHandler" />
</sectionGroup>
</configSections>
<IBM.WMQ>
<dnetxa>
<add key="ConnectionName" value=""/>
<add key="ChannelName" value="" />
<add key="QueueManagerName" value="" />
<add key="UserId" value="" />
<add key="SecurityExit" value="" />
<add key="SecurityExitUserData" value = "">
</dnetxa>
</dnetxa>
</configuration>
```

WmqDotNetXAMonitor Uygulama günlüğü
Monitor Application, Monitor 'un ilerleme durumunu ve işlem kurtarma durumunu günlüğe kaydetmek için uygulama dizininde bir günlük kütüğü yaratır. Günlüğe kaydetme işlemi, kurtarma işlemi yürütülmekte olan yürürlükteki kuyruk yöneticisini göstermek için bağlantı adı ve kanal ayrıntılarıyla başlar.

Kurtarma işlemi başladıktan sonra, hareket kurtarma iletisinin MessageId , tamamlanmamış işlemin TransactionId ve Transaction Manager Eşgüdümleme başına işlemin gerçek sonucunun günlüğe kaydedileceği bir kez.

Örnek günlük dosyası:

```
Time|ProcessId|ThreadId|WMQ .NET XA Recovery Monitor, Running now for
ConnectionName:xxxx, Time|ProcessId|ThreadId|Channel=xxxx
Time|ProcessId|ThreadId|Current QueueDepth = n
```

```
Time|ProcessId|ThreadId|Current MessageId = xxxx
Time|ProcessId|ThreadId|Current Incomplete Transaction being recovered = xxxxx
Time|ProcessId|ThreadId|Actual Outcome of the transaction(as per DTC)= Commit/Rollback
Time|ProcessId|ThreadId|Recovery Completed for TransactionId= xxxxx
Time|ProcessId|ThreadId|Current QueueDepth = n
Time|ProcessId|ThreadId|Current MessageId = xxxx
Time|ProcessId|ThreadId|Current Incomplete Transaction being recovered = xxxxx
Time|ProcessId|ThreadId|Actual Outcome of the transaction(as per DTC)= Commit/Rollback
Time|ProcessId|ThreadId| Recovery Completed for TransactionId= xxxxx
```

.NET için WebSphere MQ sınıflarının kullanılması

Bu konu grubunda, .NET kuruluşu için WebSphere MQ sınıflarınızı ve kendi programlarınızı nasıl çalıştıracağını doğrulamak üzere örnek programları çalıştırmak için sisteminizi nasıl yapılandıracağını ele alan konular açıklanmaktadır.

Kuyruk yöneticinizin TCP/IP istemci bağlantılarını kabul etmek için yapılandırılması

Bir kuyruk yöneticisini istemcilerden gelen bağlantı isteklerini kabul edecek şekilde yapılandırmak için:

1. Bir sunucu bağlantı kanalı tanımlayın:
   a. Kuyruk yöneticisini başlatın.
   b. NET.CHANNEL[3]:

```
DEF CHL('NET.CHANNEL') CHLTYPE(SVRCONN) TRPTYPE(TCP) MCAUSER(' ') +
DESCR('Sample channel for WebSphere MQ classes for .NET')
```

2. Dinleyici başlatma:

```
runmqlsr -t tcp [-m qmnqme] [-p portnum]
```

   Not: Köşeli ayraçlar isteğe bağlı değiştirgeleri belirtir; varsayılan kuyruk yöneticisi için *qmname* gerekli değildir ve varsayılan değer (1414) kullanıyorsanız kapı numarası *portnum* gerekli değildir.

Örnek Uygulamalar

Kendi .NET uygulamalarınızı çalıştırmak için, örnek uygulamalar yerine uygulama adınızı yerine koyarak, doğrulama programlarına ilişkin yönergeleri kullanın.

Beş örnek uygulama sağlanır:

- Bir put iletisi uygulaması
- İleti alma uygulaması
- 'merhaba dünya' uygulaması
- Bir yayınlama/abone olma uygulaması
- İleti özelliklerini kullanan bir uygulama

Bu örnek uygulamaların tümü C# dilinde sağlanır ve bazıları C++ dilinde ve Visual Basic 'te de sağlanır. Uygulamaları .NET tarafından desteklenen herhangi bir dilde yazabilirsiniz.

"put message" program SPUT (nmqsput.cs, mmqsput.cpp, vmqsput.vb)
Bu program, adı belirtilen bir kuyruğa nasıl ileti konacağını gösterir. Programda üç parametre vardır:

- Kuyruğun adı (gerekli), örneğin, SYSTEM.DEFAULT.LOCAL.QUEUE
- Kuyruk yöneticisinin adı (isteğe bağlı)
- Bir kanalın tanımlaması (isteğe bağlı); örneğin, SYSTEM.DEF.SVRCONN/TCP/hostname(1414)

[3] Bu örnekte, güvenlik etkilerini göz önünde bulundurmuyoruz. Bir üretim sistemi için, SSL ya da güvenlik çıkışı kullanmayı düşünün. Daha fazla bilgi için bkz. Durumu .

Kuyruk yöneticisi adı verilmezse, kuyruk yöneticisi varsayılan olarak varsayılan yerel kuyruk yöneticisine varsayılan değer olarak ayarlanır. Bir kanal tanımlandıysa, bu, MQSERVER ortam değişkeniyle aynı biçime sahiptir.

"İleti al" program SGET (nmqsget.cs, mmqsget.cpp, vmqsget.vb)
Bu program, adı belirtilen kuyruktan nasıl ileti alacağını gösterir. Programda üç parametre vardır:

- Kuyruğun adı (gerekli), örneğin, SYSTEM.DEFAULT.LOCAL.QUEUE
- Kuyruk yöneticisinin adı (isteğe bağlı)
- Bir kanalın tanımlaması (isteğe bağlı); örneğin, SYSTEM.DEF.SVRCONN/TCP/hostname(1414)

Kuyruk yöneticisi adı verilmezse, kuyruk yöneticisi varsayılan olarak varsayılan yerel kuyruk yöneticisine varsayılan değer olarak ayarlanır. Bir kanal tanımlandıysa, bu, MQSERVER ortam değişkeniyle aynı biçime sahiptir.

"Merhaba Dünya" programı (nmqwrld.cs, mmqwrld.cpp, vmqwrld.vb)
Bu program, bir iletinin nasıl yerleştireceğini ve nasıl alacağını gösterir. Programda üç parametre vardır:

- Bir kuyruğun adı (isteğe bağlı); örneğin, SYSTEM.DEFAULT.LOCAL.QUEUE YA DA SYSTEM.DEFAULT.MODEL.QUEUE
- Kuyruk yöneticisinin adı (isteğe bağlı)
- Bir kanal tanımlaması (isteğe bağlı); örneğin, SYSTEM.DEF.SVRCONN/TCP/hostname(1414)

Kuyruk adı verilmezse, ad varsayılan olarak SYSTEM.DEFAULT.LOCAL.QUEUE. Kuyruk yöneticisi adı verilmezse, kuyruk yöneticisi varsayılan olarak varsayılan yerel kuyruk yöneticisine varsayılan değer olarak ayarlanır.

"Publish/subscreen" programı (MQPubSubSample.cs)
Bu program, WebSphere MQ yayınlama/abone olma biçiminin nasıl kullanılacağını gösterir. Yalnızca C# içinde sağlanır. Programın iki değiştirgesi vardır:

- Kuyruk yöneticisinin adı (isteğe bağlı)
- Bir kanal tanımlaması (isteğe bağlı)

"İleti özellikleri" programı (MQMessagePropertiesSample.cs)
Bu program ileti özelliklerinin nasıl kullanılacağını gösterir. Yalnızca C# içinde sağlanır. Programın iki değiştirgesi vardır:

- Kuyruk yöneticisinin adı (isteğe bağlı)
- Bir kanal tanımlaması (isteğe bağlı)

Bu uygulamaları derleyerek ve çalıştırarak kuruluşunuzu doğrulayabilirsiniz.

Örnek uygulamalar, yazıldığı dile göre, aşağıdaki konumlara kurulur. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

C#

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\nmqswrld.cs

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\nmqsput.cs

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\nmqsget.cs

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\MQPubSubSample.cs

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\MQMessagePropertiesSample.cs

Yönetilen C++

MQ_INSTALLATION_PATH\Tools\dotnet\samples\mcp\mmqswrld.cpp

MQ_INSTALLATION_PATH\Tools\dotnet\samples\mcp\mmqsput.cpp

MQ_INSTALLATION_PATH\Tools\dotnet\samples\mcp\mmqsget.cpp

Visual Basic

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\vmqswrld.vb

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\vmqsput.vb

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\vmqsget.vb

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\xmqswrld.vb

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\xmqsput.vb

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\xmqsget.vb

Örnek uygulamaları oluşturmak için her dil için bir toplu iş dosyası sağlanmıştır.

C#

MQ_INSTALLATION_PATH\Tools\dotnet\samples\cs\bldcssamp.bat

bldcssamp.bat dosyası, bu örnek programı oluşturmak için gerekli olan her örnek için bir çizgi içerir.

```
csc /t:exe /r:System.dll /r:amqmdnet.dll /lib:MQ_INSTALLATION_PATH\bin
/out:nmqwrld.exe nmqwrld.cs
```

Yönetilen C++

MQ_INSTALLATION_PATH\Tools\dotnet\samples\mcp\bldmcpsamp.bat

bldmcpsamp.bat dosyası, bu örnek programı oluşturmak için gerekli olan her örnek için bir çizgi içerir:

```
cl /clr:oldsyntax MQ_INSTALLATION_PATH\bin mmqwrld.cpp
```

Bu uygulamaları Microsoft Visual Studio 2003/.NET SDKv1.1, derleme komutunu değiştirin:

```
cl /clr:oldsyntax MQ_INSTALLATION_PATH\bin mmqwrld.cpp
```

şu ürünü geçir

```
cl /clr MQ_INSTALLATION_PATH\bin mmqwrld.cpp
```

Visual Basic

MQ_INSTALLATION_PATH\Tools\dotnet\samples\vb\bldvbsamp.bat

bldvbsamp.bat dosyası, bu örnek programı oluşturmak için gerekli olan her örnek için bir çizgi içerir:

```
vbc /r:System.dll /r:MQ_INSTALLATION_PATH\bin\amqmdnet.dll /out:vmqwrld.exe vmqwrld.vb
```

WebSphere MQ .NET problemlerinin çözülmesi

Bir program başarıyla tamamlanmazsa, örnek uygulamalarından birini çalıştırın ve tanılama iletilerinde verilen önerileri izleyin.

Bu örnek uygulamalar ".NET için WebSphere MQ sınıflarının kullanılması" sayfa 549içinde açıklanmıştır.

Sorunlar devam ederse ve IBM hizmet ekibiyle iletişim kurmanız gerekiyorsa, izleme olanağını açmanız istenebilir.

Örnek uygulamanın izlenmesi

Trace olanağının kullanılmasına ilişkin yönergeler için "WebSphere MQ .NET programlarının izlenmesi" sayfa 571belgesine bakın.

hata iletileri

Aşağıdaki ortak hata iletisini görebilirsiniz:

'System.IO.FileNotFoundException' bilinmeyen modülde oluştu
Bu hata amqmdnet.dll ya da amqmdxcs.dlliçin oluşursa, her ikisinin de 'Global Assembly Cache' içinde kayıtlı olduğunu doğrulayın ya da amqmdnet.dll ve amqmdxcs.dll yapıbirimlerini işaret eden bir yapılandırma dosyası oluşturun. .NET çerçevesinin bir parçası olarak sağlanan mscorcfg.msckomutunu kullanarak montaj önbelleğinin içeriğini inceleyebilir ve değiştirebilirsiniz.

WebSphere MQ kurulu olduğunda .NET çerçevesi kullanılamıyorsa, sınıflar genel derleme önbelleğine kayıtlı olmayabilir. Komutu kullanarak kayıt işlemini el ile yeniden çalıştırabilirsiniz.

```
amqidnet -c MQ_INSTALLATION_PATH\bin\amqidotn.txt -l logfile.txt
```

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Bu kuruluma ilişkin bilgiler, belirtilen günlük dosyasına yazılır (bu örnek`logfile.txt`).

WebSphere MQ .NET programlarının yazılması ve konuşlandırılması

.NET için WebSphere MQ sınıflarını WebSphere MQ kuyruklarına erişmek üzere kullanmak için, programları, iletileri içeren çağrılar içeren .NET tarafından desteklenen herhangi bir dilde yazar ve WebSphere MQ kuyruklarından ileti alır.

WebSphere MQ belgeleri yalnızca C#, C++ ve Visual Basic dillerinde bilgi içerir.

This collection of topics provides information to assist with writing applications to interact with WebSphere MQ systems. Tek tek sınıflara ilişkin ayrıntılar için WebSphere MQ .NET sınıflarına ve arabirimlerinebakın.

Bağlantı farkları

WebSphere MQ .NET için kullanma şekliniz, kullanmak istediğiniz bağlantı kiplerine bazı bağımlılıklar içerir.

Yönetilen istemci bağlantıları
Yönetilen istemci olarak .NET için WebSphere MQ sınıfları kullanıldığında, standart bir WebSphere MQ MQI istemcisinden bir dizi farklılıklar vardır.

Yönetilen bir istemci için aşağıdaki özellikler kullanılamaz:

- Kanal sıkıştırması
- SSL desteği
- Kanal çıkışı zincirleme

Bu özellikleri yönetilen bir istemciyle kullanmaya çalışırsanız, bu bir MQException döndürür. Hata, bağlantının istemci ucunda saptanırsa, MQRC_ENVIRONMENT_ERROR neden kodunu kullanır. Sunucu ucunda saptanırsa, sunucu tarafından döndürülen neden kodu kullanılır.

Yönetilmeyen bir istemci için yazılan kanal çıkışları işe yaramaz. Yönetilen istemci için özel olarak yeni çıkışlar yazmalısınız. CCDT (istemci kanal tanımlama çizelgesinde (CCDT) geçersiz kanal çıkışı belirtilmemesine dikkat edin.

Yönetilen kanal çıkışısının adı en çok 999 karakter uzunluğunda olabilir. Ancak, kanal çıkış adını belirtmek için CCDT ' yi kullanırsanız, bu değer 128 karakterle sınırlıdır.

İletişim yalnızca TCP/IP üzerinde desteklenir.

endmqm komutunu kullanarak bir kuyruk yöneticisini durdurduğunuzda, .NET yönetilen istemcisine bir sunucu bağlantısı kanalı, sunucu bağlantısı kanallarının diğer istemcilere daha uzun sürmesini sağlar.

Yönetilen WebSphere MQ sorun tanılama programlarını kullanmak için *NMQ_MQ_LIB* ' i `yönetilen` olarak ayarladıysanız, **strmqtrc** komutunun -i, -p, -s, -b ya da -c parametrelerinin hiçbiri desteklenmektedir.

XA hareketleri kullanan yönetilen bir .NET uygulaması, bir z/OS kuyruk yöneticisiyle çalışmaz. Yönetilen. Bir z/OS kuyruk yöneticisine bağlanmaya çalışan bir ağ istemcisi bir hatayla başarısız oluyor, MQRICE çağrısında MQRC_UOW_ENLISTMENT_ERROR (mqrc=2354). Ancak, XA hareketleri kullanan bir yönetilen .NET uygulaması dağıtılmış kuyruk yöneticisiyle çalışır.

Kullanılacak bağlantı tipini tanımlama

Bağlantı tipi, bağlantı adı, kanal adı, uyarlama değeri NMQ_MQ_LIB ve özellik MQC.TRANSPORT_PROPERTY.

Bağlantı adını aşağıdaki gibi belirtebilirsiniz:

- Bir MQQueueManager oluşturucusuna açık bir şekilde:

```
public MQQueueManager(String queueManagerName, MQLONG Options, string Channel,
string ConnName)
```

```
public MQQueueManager(String queueManagerName, string Channel, string ConnName)
```

- By setting the properties MQC.HOST_NAME_PROPERTY and, optionally, MQC.PORT_PROPERTY in a hashtable entry on an MQQueueManager constructor:

```
public MQQueueManager(String queueManagerName, Hashtable properties)
```

- Açık MQEnvironment değerleri olarak

```
MQEnvironment.Hostname
```

  `MQEnvironment.Port`(isteğe bağlı).
- By setting the properties MQC.HOST_NAME_PROPERTY and, optionally, MQC.PORT_PROPERTY in the MQEnvironment.properties hashtable.

Kanal adını aşağıdaki gibi belirtebilirsiniz:

- Bir MQQueueManager oluşturucusuna açık bir şekilde:

```
public MQQueueManager(String queueManagerName, MQLONG Options, string Channel,
string ConnName)
```

```
public MQQueueManager(String queueManagerName, string Channel, string ConnName)
```

- MQC.CHANNEL_PROPERTY , bir MQQueueManager oluşturucusuna ilişkin hashtable girişlerinde:

```
public MQQueueManager(String queueManagerName, Hashtable properties)
```

- Açık bir MQEnvironment değeri olarak

```
MQEnvironment.Channel
```

- MQC.CHANNEL_PROPERTY , MQEnvironment.properties hashtable 'ında.

İletim özelliğini aşağıdaki gibi belirtebilirsiniz:

- MQC.TRANSPORT_PROPERTY , bir MQQueueManager oluşturucusuna ilişkin hashtable girişlerinde:

```
public MQQueueManager(String queueManagerName, Hashtable properties)
```

- MQC.TRANSPORT_PROPERTY , MQEnvironment.properties hashtable.

Aşağıdaki değerlerden birini kullanarak, gerek duyduğunuz bağlantı tipini seçin:

MQC.TRANSPORT_MQSERIES_BINDINGS -sunucu olarak bağlantı kur
MQC.TRANSPORT_MQSERIES_CLIENT -XA dışı istemci olarak bağlantı kurun
MQC.TRANSPORT_MQSERIES_XACLIENT -XA istemcisi olarak bağlan
MQC.TRANSPORT_MQSERIES_MANAGED -XA dışı yönetilen istemci olarak bağlanır

NMQ_MQ_LIB uyarlama değerini, aşağıdaki tabloda gösterildiği gibi bağlantı tipini açık bir şekilde seçmek için ayarlayabilirsiniz.

NMQ_MQ_LIB değeri	**Bağlantı tipi**
mqic.dll	XA dışı bir istemci olarak bağlan
mqicxa.dll	XA istemcisi olarak bağlan
mqm.dll	Sunucu olarak ya da XA dışı bir istemci olarak bağlan
yönetilen	XA dışı yönetilen bir istemci olarak bağlan
Not: mqic32.dll ve mqic32xa.dll değerleri, daha önceki yayınlarla uyumluluk sağlamak için mqic.dll ve mqicxa.dll eşanlamlıları olarak kabul edilir. Ancak, mqm.dll ve mqm.pdb , istemci paketinin yalnızca 7.1 sürümünden itibaren bir parçalarıdır.	

Ortamınızda kullanılamayan bir bağlantı tipini seçerseniz, örneğin mqic32xa.dll değerini belirtmiş ve XA desteği yoksa, WebSphere MQ .NET bir kural dışı durum yayınlıyor.

NMQ_MQ_LIB ' yi "yönetilen" olarak ayarlamak, istemcinin yönetilen WebSphere MQ sorun tanılama sınamalarını, .NET veri dönüştürmesini ve diğer yönetilen alt düzey WebSphere MQ işlevlerini kullanmasına neden olur.

NMQ_MQ_LIB için diğer tüm değerler, .NET işleminin yönetilmeyen WebSphere MQ sorun tanılama sınamalarını ve veri dönüştürme işlemini ve yönetilmeyen diğer düşük düzeyli WebSphere MQ işlevlerini (bir WebSphere MQ MQI istemcisi ya da sunucusu sistemde kurulu olduğu varsayılarak) kullanmasını sağlar.

WebSphere MQ .NET, bağlantı tipini aşağıdaki gibi seçer:

1. MQC.TRANSPORT_PROPERTY belirtildi, MQC.TRANSPORT_PROPERTY.

   Ancak, MQC.TRANSPORT_PROPERTY - MQC.TRANSPORT_MQSERIES_MANAGED , istemci işleminin yönetiliyor olduğunu garanti etmez. Bu ayara rağmen, istemci aşağıdaki durumlarda yönetilmiyor:

   - Süreçteki başka bir iş parçacığı MQC.TRANSPORT_PROPERTY , MQC.TRANSPORT_MQSERIES_MANAGED.
   - NMQ_MQ_LIB "yönetilen" olarak ayarlanmazsa, sorun tanılama sınamaları, veri dönüştürmesi ve diğer düşük düzeyli işlevler tam olarak yönetilemez (sistemde bir WebSphere MQ MQI istemcisi ya da sunucusu kurulu olduğu varsayılarak).
2. Bir bağlantı adı, kanal adı olmadan ya da bağlantı adı olmadan bir kanal adı belirlendiyse, bir hata oluşur.
3. Hem bir bağlantı adı, hem de kanal adı belirtilmişse:
   - NMQ_MQ_LIB, mqic32xa.dllolarak ayarlandıysa, XA istemcisi olarak bağlanır.
   - Yönetilen olarak NMQ_MQ_LIB ayarlanmışsa, yönetilen istemci olarak bağlanır.
   - Tersi durumda, XA istemcisi olmayan bir istemci olarak bağlanır.
4. NMQ_MQ_LIB belirtilirse, bu, NMQ_MQ_LIB değerine göre bağlanır.
5. Bir WebSphere MQ sunucusu kuruluysa, sunucu bir sunucu olarak bağlanır.
6. Bir WebSphere MQ MQI istemcisi kuruluysa, XA istemcisi olmayan bir istemci olarak bağlanır.
7. Tersi durumda, yönetilen istemci olarak bağlanır.

.NET için WebSphere MQ sınıflarına ilişkin yapılandırma dosyaları

Bir .NET istemci uygulaması bir WebSphere MQ MQI istemcisi yapılanış kütüğünü kullanabilir ve yönetilen bağlantı tipini kullanıyorsanız, bir .NET uygulaması yapılandırma dosyası kullanılabilir. Uygulama yapılandırma dosyasındaki ayarlar önceliğe sahiptir.

İstemci yapılandırma dosyası

.NET istemci uygulaması için bir WebSphere MQ sınıfları, diğer herhangi bir WebSphere MQ MQI istemciyle aynı şekilde bir istemci yapılanış kütüğünü kullanabilir. Bu dosya genellikle mqclient.iniolarak adlandırılır, ancak farklı bir dosya adı belirleyebilirsiniz. İstemci yapılandırma dosyası hakkında daha fazla bilgi için bkz. Yapılandırma dosyası WebSphere MQ MQI istemcisi yapılandırma dosyası kullanarak istemci yapılandırılması.

Bir WebSphere MQ MQI istemcisi yapılanış dosyasındaki yalnızca aşağıdaki öznitelikler, .NET için WebSphere MQ sınıflarıyla ilgilidir. Diğer öznitelikleri belirlerseniz, bu bir etki göstermez.

Stanza	**Öznitelik**
Kanallar	CCSID
Kanallar	ChannelDefinitionDizini
Kanallar	ChannelDefinitionDosyası
Kanallar	ServerConnectionParms
ClientExitYolu	ExitsDefaultYolu
ClientExitYolu	ExitsDefaultPath64
MessageBuffer	MaximumSize
MessageBuffer	PurgeTime
MessageBuffer	UpdatePercentage
TCP	ClntRcvBufSize
TCP	ClntSndBufSize
TCP	IPAddressVersion
TCP	KeepAlive

Uygun ortam değişkenini kullanarak bu özniteliklerden herhangi birini geçersiz kılabilirsiniz.

Uygulama yapılandırma dosyası

Yönetilen bağlantı tipiyle çalıştırıyorsanız, .NET uygulaması yapılandırma dosyasını kullanarak WebSphere MQ istemcisi yapılandırma dosyasını ve eşdeğer ortam değişkenlerini de geçersiz kılabilirsiniz.

.NET uygulaması yapılandırma dosyası ayarları yalnızca, yönetilen bağlantı tipiyle çalıştırılırken ya da diğer bağlantı tipleri için yoksayılır.

.NET uygulaması yapılandırma dosyası ve biçimi Microsoft tarafından .NET çerçevesi içinde genel kullanım için tanımlansa da, bu belgede sözü edilen bölüm adları, anahtarlar ve değerler Websphere MQ' ya özgütür.

.NET uygulaması yapılandırma dosyasının biçimi, *kısımlar*sayısıdır. Her bir bölüm bir ya da daha çok *anahtar*içerir ve her anahtarın ilişkili bir *değer*ivardır. Aşağıdaki örnekte, TCP/IP KeepAlive özelliğini denetlemek için bir .NET uygulaması yapılandırma dosyasında kullanılan kısımlar, anahtarlar ve değerler gösterilmektedir:

```
<configuration>
  <configSections>
```

```
    <section name="TCP" type="System.Configuration.NameValueSectionHandler"/>
  </configSections>
  <TCP>
    <add key="KeepAlive" value="true"></add>
  </TCP>
<configuration>
```

.NET uygulaması yapılandırma dosyası kısım adlarında ve anahtarlarında kullanılan anahtar sözcükler, istemci yapılandırma dosyasında tanımlı olan Stanzalar ve Attributes için anahtar sözcüklerle tam olarak eşleşir.

Ek bilgi için Microsoft belgelerinize bakın.

Örnek kod parçası

Aşağıdaki C# kod parçası, üç işlem gerçekleştiren bir uygulamayı gösterir:

1. Kuyruk yöneticisine bağlan
2. SYSTEM.DEFAULT.LOCAL.QUEUE
3. İletiyi geri al

Bağlantı tipinin nasıl değiştirileceğini de gösterir.

```
// ============================================================================
// Licensed Materials - Property of IBM
// 5724-H72
// (c) Copyright IBM Corp. 2003, 2024
// ============================================================================
using System;
using System.Collections;

using IBM.WMQ;

class MQSample
{
  // The type of connection to use, this can be:-
  // MQC.TRANSPORT_MQSERIES_BINDINGS for a server connection.
  // MQC.TRANSPORT_MQSERIES_CLIENT for a non-XA client connection
  // MQC.TRANSPORT_MQSERIES_XACLIENT for an XA client connection
  // MQC.TRANSPORT_MQSERIES_MANAGED for a managed client connection
  const String connectionType = MQC.TRANSPORT_MQSERIES_CLIENT;

  // Define the name of the queue manager to use (applies to all connections)
  const String qManager = "your_Q_manager";

  // Define the name of your host connection (applies to client connections only)
  const String hostName = "your_hostname";

  // Define the name of the channel to use (applies to client connections only)
  const String channel = "your_channelname";



  /// <summary>
  /// Initialise the connection properties for the connection type requested
  /// </summary>
  /// <param name="connectionType">One of the MQC.TRANSPORT_MQSERIES_ values</param>
  static Hashtable init(String connectionType)
  {
    Hashtable connectionProperties = new Hashtable();

    // Add the connection type
    connectionProperties.Add(MQC.TRANSPORT_PROPERTY, connectionType);

    // Set up the rest of the connection properties, based on the
    // connection type requested
    switch(connectionType)
    {
      case MQC.TRANSPORT_MQSERIES_BINDINGS:
        break;
      case MQC.TRANSPORT_MQSERIES_CLIENT:
      case MQC.TRANSPORT_MQSERIES_XACLIENT:
      case MQC.TRANSPORT_MQSERIES_MANAGED:
        connectionProperties.Add(MQC.HOST_NAME_PROPERTY, hostName);
        connectionProperties.Add(MQC.CHANNEL_PROPERTY, channel);
```

```
        break;
    }

    return connectionProperties;
  }
  /// <summary>
  /// The main entry point for the application.
  /// </summary>
  [STAThread]
  static int Main(string[] args)
  {
    try
    {
      Hashtable connectionProperties = init(connectionType);

      // Create a connection to the queue manager using the connection
      // properties just defined
      MQQueueManager qMgr = new MQQueueManager(qManager, connectionProperties);

      // Set up the options on the queue we want to open
      int openOptions = MQC.MQOO_INPUT_AS_Q_DEF | MQC.MQOO_OUTPUT;

      // Now specify the queue that we want to open,and the open options
      MQQueue system_default_local_queue =
        qMgr.AccessQueue("SYSTEM.DEFAULT.LOCAL.QUEUE", openOptions);

      // Define a WebSphere MQ message, writing some text in UTF format
      MQMessage hello_world = new MQMessage();
      hello_world.WriteUTF("Hello World!");

      // Specify the message options
      MQPutMessageOptions pmo = new MQPutMessageOptions(); // accept the defaults,
                                                           // same as MQPMO_DEFAULT

      // Put the message on the queue
      system_default_local_queue.Put(hello_world, pmo);



      // Get the message back again

      // First define a WebSphere MQ message buffer to receive the message
      MQMessage retrievedMessage =new MQMessage();
      retrievedMessage.MessageId =hello_world.MessageId;

      // Set the get message options
      MQGetMessageOptions gmo =new MQGetMessageOptions(); //accept the defaults
                                                          //same as MQGMO_DEFAULT

      // Get the message off the queue
      system_default_local_queue.Get(retrievedMessage,gmo);

      // Prove we have the message by displaying the UTF message text
      String msgText = retrievedMessage.ReadUTF();
      Console.WriteLine("The message is: {0}", msgText);

      // Close the queue
      system_default_local_queue.Close();

      // Disconnect from the queue manager
      qMgr.Disconnect();
    }

    //If an error has occurred,try to identify what went wrong.

    //Was it a WebSphere MQ error?
    catch (MQException ex)
    {
      Console.WriteLine("A WebSphere MQ error occurred: {0}", ex.ToString());
    }

    catch (System.Exception ex)
    {
      Console.WriteLine("A System error occurred: {0}", ex.ToString());
    }

    return 0;
  }//end of start
}//end of sample
```

Kuyruk yöneticilerine ilişkin işlemler

Bu kısımda, .NET için WebSphere MQ sınıflarını kullanan bir kuyruk yöneticiyle bağlantı kurulacağı ve kuyruk yöneticisinden bağlantı kesileceği açıklanmaktadır.

WebSphere MQ ortamını ayarlama

Bir kuyruk yöneticisine bağlanmak için istemci bağlantısını kullanmadan önce, WebSphere MQ ortamını ayarlamalısınız.

Not: Sunucu bağ tanımları kipinde .NET için WebSphere MQ sınıfları kullanılırken bu adım gerekli değildir.

.NET programlama arabirimi, NMQ_MQ_LIB uyarlama değerini kullanmanızı, aynı zamanda bir MQEnvironment sınıfı da içermenizi sağlar. Bu sınıf, aşağıdaki listede yer alan, bağlantı girişimi sırasında kullanılacak ayrıntıları belirtmenizi sağlar:

- Kanal adı
- Anasistem adı
- Kapı numarası
- Kanal çıkışları
- SSL parametreleri
- Kullanıcı kimliği ve parola

MQEnvironment sınıflarıyla ilgili tam bilgi için bakınız: MQEnvironment .NET class

Kanal adını ve anasistem adını belirtmek için aşağıdaki kodu kullanın:

```
MQEnvironment.Hostname = "host.domain.com";
MQEnvironment.Channel  = "client.channel";
```

Varsayılan olarak, istemciler 1414 numaralı kapıdaki bir WebSphere MQ dinleyicisine bağlanmayı deneyer. Farklı bir kapı belirtmek için şu kodu kullanın:

```
MQEnvironment.Port = nnnn;
```

Kuyruk yöneticiyle bağlantı kuruluyor

Artık MQQueueManager sınıfının yeni bir örneğini oluşturarak bir kuyruk yöneticisine bağlanmaya hazırsınız:

```
MQQueueManager queueManager = new MQQueueManager("qMgrName");
```

Kuyruk yöneticisinden bağlantıyı kesmek için, kuyruk yöneticisinden `Disconnect` yöntemini çağırın:

```
queueManager.Disconnect();
```

Kuyruk yöneticisine bağlanma girişimi sırasında kuyruk yöneticisine ilişkin sorgulama (`inq`) yetkinizin olması gerekir. Sorgu yetkisi olmadan, bağlantı kurma girişimi başarısız olur.

`Disconnect` yöntemini çağırırsanız, o kuyruk yöneticisi aracılığıyla eriştiğiniz tüm açık kuyruklar ve işlemler kapatılır. Ancak, bu kaynakları kullanmayı bitirdiğinizde, bu kaynakları açık bir şekilde kapatmak için iyi bir programlama uygulamasıdır. Kaynakları kapatmak için, her kaynakla ilişkilendirilmiş nesneyle ilgili `Close` yöntemini kullanın.

Kuyruk yöneticisiyle ilgili `Commit` ve `Backout` yöntemleri, yordamsal arabirimle birlikte kullanılan MQCMIT ve MQBACK çağrılarını değiştirir.

Kuyruklara ve konulara erişilmesi

You can access queues and topics using methods of MQQueueManager or appropriate constructors.

Kuyruklara erişmek için, MQQueueManager sınıfının yöntemlerini kullanın. MQOD (nesne tanımlayıcı yapısı), bu yöntemlerin parametrelerine daraltılır. Örneğin, queueManageradlı bir MQQueueManager nesnesiyle gösterilen kuyruk yöneticileninde bir kuyruk açmak için aşağıdaki kodu kullanın:

```
MQQueue queue = queueManager.AccessQueue("qName",
                                         MQC.MQOO_OUTPUT,
                                         "qMgrName",
                                         "dynamicQName",
                                         "altUserId");
```

seçenekler parametresi, MQOPEN çağrısındaki Options parametresiyle aynıdır.

AccessQueue yöntemi, MQQueue sınıfını yeni bir nesne döndürür.

Kuyruğu kullanmayı bitirdiğinizde, aşağıdaki örnekteki gibi kapatmak için Close () yöntemini kullanın:

```
queue.Close();
```

WebSphere MQ .NET ile, MQQueue oluşturucusunu kullanarak bir kuyruk da yaratabilirsiniz. Parametreler, kullanılacak örnek MQQueueManager nesnesini belirten bir kuyruk yöneticisi değiştirgesinin eklenmesiyle, tam olarak accessQueue yöntemi ile aynıdır. Örneğin:

```
MQQueue queue = new MQQueue(queueManager,
                            "qName",
                            MQC.MQOO_OUTPUT,
                            "qMgrName",
                            "dynamicQName",
                            "altUserId");
```

Bu şekilde bir kuyruk nesnesi oluşturulması, kendi MQQueue alt sınıflarınızı yazmanızı sağlar.

Benzer şekilde, konulara MQQueueManager sınıfının yöntemlerini kullanarak da erişebilirsiniz. Bir konuyu açmak için bir AccessTopic() yöntemini kullanın. Bu, sınıf MQTopic 'in yeni bir nesnesini döndürür. Konuyu kullanmayı bitirdiğinizde, bu konuyu kapatmak için MQTopic 'in Close () yöntemini kullanın.

Ayrıca, bir MQTopic oluşturucusu kullanarak bir konu da yaratabilirsiniz. Konular için bir dizi oluşturucular vardır; daha fazla bilgi için bkz. MQTopic .NET sınıfı.

İletilerin işlenmesi

İletiler, kuyruğun ya da konu sınıflarının yöntemleri kullanılarak işlenir. Yeni bir ileti oluşturmak için yeni bir MQMessageobject yaratın.

MQQueue ya da MQTopic sınıfının put () yöntemini kullanarak iletileri ya da konuları kuyruğa ya da konuya koyun. MQQueue ya da MQTopic sınıfının Get () yöntemini kullanarak kuyruklardan ya da konulardan ileti alın. Yordamsal arabirimden farklı olarak, burada MQPUT ve MQGET byte dizilerinin bayt dizileri, .NET için WebSphere MQ sınıfları put ve get eşgörünümlerini almak için. MQMessage sınıfı, bu iletiyi açıklayan tüm MQMD (ileti tanımlayıcı) değiştirgeleriyle birlikte gerçek ileti verilerini içeren veri arabelleğinden sarkılanır.

Yeni bir ileti oluşturmak için, MQMessage sınıfının yeni bir eşgörünümünü yaratın ve ileti arabelleğindeki verileri yerleştirmek için WriteXXX yöntemlerini kullanın.

Yeni ileti eşgörünümü yaratıldığında, tüm MQMD parametreleri otomatik olarak varsayılan değerlerine ayarlanır (MQMD için ilk değerler ve dil bildirimleri ' da tanımlandığı gibi). MQQueue yönteminin put () yöntemi, değiştirge olarak MQPutMessageSeçenekleri sınıfının bir örneğini de alır. Bu sınıf MQPMO yapısını temsil eder. Aşağıdaki örnek bir ileti yaratır ve bunu bir kuyruğa koyar:

```
// Build a new message containing my age followed by my name
MQMessage myMessage = new MQMessage();
myMessage.WriteInt(25);
```

```
String name = "Charlie Jordan";
myMessage.WriteUTF(name);

// Use the default put message options...
MQPutMessageOptions pmo = new MQPutMessageOptions();

// put the message!
queue.Put(myMessage,pmo);
```

MQQueue olanağının get () yöntemi, kuyruktan yeni alınan iletiyi gösteren yeni bir MQMessage eşgörünümü döndürür. Ayrıca, parametre olarak MQGetMessageOptions sınıfının bir eşgörünümünü alır. Bu sınıf MQGMO yapısını temsil eder.

Get () yöntemi, gelen iletiye sığması için iç arabelleğindeki büyüklüğü otomatik olarak ayarlandığından, bir ileti büyüklüğü üst sınırı belirtmeniz gerekmez. Döndürülen iletteki verilere erişmek için MQMessage sınıfının ReadXXX yöntemlerini kullanın.

Aşağıdaki örnekte, kuyruktan iletinin nasıl alacalacağı gösterilmektedir:

```
// Get a message from the queue
MQMessage theMessage    = new MQMessage();
MQGetMessageOptions gmo = new MQGetMessageOptions();
queue.Get(theMessage,gmo);  // has default values

// Extract the message data
int age = theMessage.ReadInt();
String name1 = theMessage.ReadUTF();
```

kodlama üye değişkenini ayarlayarak, okuma ve yazma yöntemlerinin kullandığı sayı biçimini değiştirebilirsiniz.

characterSet adlı üye değişkenini ayarlayarak dizgileri okumak ve yazmak için kullanılacak karakter kümesini değiştirebilirsiniz.

Daha fazla ayrıntı için bkz. MQMessage .NET sınıfı .

Not: MQMessage WriteUTF() yöntemi, içerdiği Unicode baytların yanı sıra dizginin uzunluğunu da otomatik olarak kodlar. İletiniz başka bir .NET programı tarafından okunacağı zaman (ReadUTF() işlevini kullanarak), dizilim bilgileri göndermenin en kolay yoludur.

İleti özelliklerinin işlenmesi

İleti özellikleri, iletileri seçmenize ya da üstbilgilerine erişmeden bir iletiyle ilgili bilgileri almanızı sağlar. MQMessage sınıfı, özellikleri almak ve ayarlamak için yöntemler içerir.

Bir uygulamanın işlenecek iletileri seçmesine izin vermek ya da MQMD ya da MQRFH2 üstbilgilerine erişmeden bir iletiyle ilgili bilgileri almak için ileti özelliklerini kullanabilirsiniz. Ayrıca, WebSphere MQ ve JMS uygulamaları arasındaki iletişimi de kolaylaştırır. WebSphere MQ' da ileti özellikleri hakkında daha fazla bilgi için bkz. İleti özellikleri.

MQMessage sınıfı, özelliğin veri tipine göre özellikleri almak ve ayarlamak için bir dizi yöntem sağlar. Alma yöntemlerinde Get * Özelliği ve set yöntemlerinin adları şu biçim kümesi * özelliğine sahiptir; burada yıldız işareti (*) aşağıdaki dizgilerden birini gösterir:

- Boole
- Byte
- Bayt
- Çift
- Kayar Noktalı Sayı
- Tamsayı
- Int2
- Int4
- Int8
- Uzun

- Nesne
- Kısa
- Dizgi

Örneğin, WebSphere MQ property myproperty (bir karakter dizgisi) almak için, `message.GetStringProperty('myproperty')` çağrısını kullanın. İsteğe bağlı olarak, WebSphere MQ ' un tamamlanacağı bir özellik tanımlayıcısını iletebilirsiniz.

Hataların işlenmesi

Handle errors arising from WebSphere MQ classes for .NET using `try` and `catch` blocks.

.NET arabirimindeki yöntemler bir tamamlanma kodu ve neden kodu döndürmez. Bunun yerine, bir WebSphere MQ çağrısından kaynaklanan tamamlanma kodu ve neden kodu her ikisi de sıfır değilse, bir kural dışı durum yayınlarlar. Bu, her bir WebSphere MQçağrısından sonra dönüş kodlarını kontrol etmek zorunda kalmamak için program mantığını basitleştirir. Programınızın hangi noktalarda hata olma olasılığına karşı karar vereceğine karar verebilirsiniz. Bu noktalarda kodunuzu `try` ve `catch` blokları ile çevrebilirsiniz. Örneğin:

```
try
{
  myQueue.Put(messageA,PutMessageOptionsA);
  myQueue.Put(messageB,PutMessageOptionsB);
}
catch (MQException ex)
{
  // This block of code is only executed if one of
  // the two put methods gave rise to a non-zero
  // completion code or reason code.
  Console.WriteLine("An error occurred during the put operation:" +
                        "CC = " + ex.CompletionCode +
                        "RC = " + ex.ReasonCode);
  Console.WriteLine("Cause exception:" + ex );
}
```

Öznitelik değerlerinin alınması ve ayarlanması

MQManagedObject, MQQueue ve MQQueueManager sınıfları, öznitelik değerlerini almanıza ve ayarlamanıza izin veren yöntemler içerir. MQQueue için, yöntemlerin yalnızca, kuyruğu açtığınızda uygun sorgu ve küme işaretlerini belirttiğinizde işe yaradığını göz önünde bulundurun.

For common attributes, the MQQueueManager and MQQueue classes inherit from a class called MQManagedObject. Bu sınıf, Sorgula () ve Set () arabirimlerini tanımlar.

Yeni işlecini kullanarak yeni bir kuyruk yöneticisi nesnesi yarattığınızda, bu nesne otomatik olarak sorgulanmak üzere açılır. When you use the AccessQueue() method to access a queue object, that object is *değil* automatically opened for either inquire or set operations, this could cause problems with some types of remote queues. Sorgulamak ve Ayarlama yöntemlerini kullanmak ve bir kuyrukta özellikleri ayarlamak için, AccessQueue() yönteminin openOptions değiştirgesinde uygun olarak sorgu ve ayar işaretlerini belirlemeniz gerekir.

Sorgu ve ayarlama yöntemleri üç parametre alır:

- seçiciler dizisi
- intAttrs dizisi
- charAttrs dizisi

Bir dizinin uzunluğu her zaman bilindiğinden, MQINQ ' da bulunan SelectorCount, IntAttrCount ve CharAttrLength değiştirgelerine gerek yoktur. Aşağıdaki örnek, bir kuyruğun nasıl bir kuyruğun üzerinde nasıl yapılır gösterileceğini göstermektedir:

```
//inquire on a queue
int [ ] selectors = new int [2] ;
int [ ] intAttrs = new int [1] ;
```

```
byte [ ] charAttrs = new byte [MQC.MQ_Q_DESC_LENGTH];
selectors [0] = MQC.MQIA_DEF_PRIORITY;
selectors [1] = MQC.MQCA_Q_DESC;
queue.Inquire(selectors,intAttrs,charAttrs);
ASCIIEncoding enc = new ASCIIEncoding();
String s1 = "";
s1 = enc.GetString(charAttrs);
```

Bu nesnelerin tüm öznitelikleri sorgulanabilir. Öznitelikler bir alt kümesi, bir nesnenin özellikleri olarak gösterilir. Nesne özniteliklerinin listesi için Nesnelerin öznitelikleribaşlıklı konuya bakın. Nesne özellikleri için, uygun sınıf tanımına bakın.

Çok iş parçacıklı programlar

.NET yürütme ortamı doğal olarak çok iş parçacıklıdır. .NET içinWebSphere MQ sınıfları, bir kuyruk yöneticisi nesnesinin birden çok iş parçacığının paylaşılmasına olanak sağlar, ancak hedef kuyruk yöneticisine tüm erişimin uyumlulaştırılmasını sağlar.

Bir kuyruk yöneticisine bağlanan ve başlatma sırasında kuyruk açan basit bir programı düşünün. Program ekranda tek bir düğme görüntüler. Bir kullanıcı bu düğmeyi tıklattığında, program kuyruktan bir ileti alır. Bu durumda, uygulamanın ilk kullanıma hazırlanması bir iş parçacığıda gerçekleşir ve düğme tuşuna yanıt olarak yürütülen kod, ayrı bir iş parçacığıda (kullanıcı arabirimi iş parçacığı) yürütülür.

WebSphere MQ .NET uygulaması, belirli bir bağlantı (MQQueueManager nesne eşgörünümü) için, hedef WebSphere MQ kuyruk yöneticisine tüm erişim erişiminin uyumlulaştırılmasını sağlar. Varsayılan davranış, bir kuyruk yöneticisine çağrı yapmak isteyen bir iş parçacığın, ilgili bağlantı için devam etmekte olan diğer tüm çağrılar tamamlanıncaya kadar engellenmiştir. Programınızdaki birden çok iş parçacığının aynı kuyruk yöneticisine eşzamanlı olarak erişmeniz gerekiyorsa, eşzamanlı erişim gerektiren her iş parçacığı için yeni bir MQQueueManager nesnesi yaratın. (Bu, her iş parçacığı için ayrı bir MQCONN çağrısı yayınlamaya eşdeğerdir.)

Varsayılan bağlantı seçenekleri MQC.MQCNO_HANDLE_SHARE_NONE ya da MQC.MQCNO_SHARE_NO_BLOCK , kuyruk yöneticisi artık uyumlulaştırılmadı.

.NET içeren bir istemci kanal tanımlama çizelgesi kullanılması

WebSphere MQiçin .NET sınıflarına sahip bir istemci kanalı tanımlama çizelgesi (CCDT) kullanabilirsiniz. CCDT ' nin yerini, yönetilen ya da yönetilmeyen bir bağlantı kullanıp kullanmamanıza bağlı olarak farklı şekillerde belirtiyorsunuz.

XA dışı ya da XA yönetilmeyen istemci bağlantısı tipi

Yönetilmeyen bir bağlantı tipiyle CCDT ' nin yerini iki şekilde belirtebilirsiniz:

- Çizelgenin yer aldığı dizini belirtmek için MQCHLLIB ortam değişkenlerini ve çizelgenin dosya adını belirtmek için MQCHLTAB ' ı kullanın.
- İstemci yapılandırma dosyası kullanılıyor. CHANNELDEFINITION kısmında, çizelgenin bulunduğu dizini belirlemek için ChannelDefinitionDizinini ve dosya adını belirtmek için ChannelDefinitiondosyasını kullanın.

Konum hem istemci yapılandırma dosyasında, hem de ortam değişkenleri kullanılarak belirtilirse, ortam değişkenleri önceliğe sahip olur. Bu özelliği, istemci yapılanış dosyasında standart bir yer belirtmek ve gerektiğinde ortam değişkenlerini kullanarak geçersiz kılmak için kullanabilirsiniz.

Yönetilen istemci bağlantısı tipi

Yönetilen bir bağlantı tipiyle CCDT ' nin yerini üç şekilde belirtebilirsiniz:

- .NET uygulaması yapılandırma dosyası kullanılıyor. CHANELS kısmında, çizelgenin bulunduğu dizini belirlemek için ChannelDefinitionDizinini ve dosya adını belirtmek için ChannelDefinitiondosyasını kullanın.
- Çizelgenin yer aldığı dizini belirtmek için MQCHLLIB ortam değişkenlerini ve çizelgenin dosya adını belirtmek için MQCHLTAB ' ı kullanın.

- İstemci yapılandırma dosyası kullanılıyor. CHANNELDEFINITION kısmında, çizelgenin bulunduğu dizini belirlemek için ChannelDefinitionDizinini ve dosya adını belirtmek için ChannelDefinitiondosyasını kullanın.

Yer, bu şekilde birden çok şekilde belirtilirse, ortam değişkenleri istemci yapılandırma dosyasına göre öncelik alır ve .NET Uygulama Yapılandırma Dosyası diğer iki yöntemden de öncelikli olarak önceliğe sahip olur. Bu özelliği, istemci yapılanış dosyasında standart bir yer belirtmek ve gerektiğinde, ortam değişkenlerini ya da uygulama yapılandırma dosyasını kullanarak geçersiz kılmak için kullanabilirsiniz.

.NET uygulaması hangi kanal tanımlamasını kullanacağını belirler

WebSphere MQ .NET istemcisi ortamında, kullanılacak kanal tanımlaması farklı şekillerde belirtilebilir. Kanal tanımlamasının birden çok belirtimi var olabilir. Bir uygulama, kanal tanımını bir ya da daha çok kaynaktan türetir.

Birden çok kanal tanımlaması varsa, kullanılan öğe aşağıdaki öncelik sırasına göre seçilir:

1. Properties specified on the MQQueueManager constructor, either explicitly or by including *MQC.CHANNEL_PROPERTY* in the properties hashtable
2. Bir özellik *MQC.CHANNEL_PROPERTY* , MQEnvironment.properties hashtable içinde
3. MQEnvironment 'daki *Kanal* özelliği
4. .NET uygulaması yapılandırma dosyası, bölüm adı KANALS, anahtar ServerConnectionParms (yalnızca yönetilen bağlantılar için geçerlidir)
5. *MQSERVER* ortam değişkeni
6. İstemci yapılandırma dosyası, stanza KANALLARI, Öznitelik ServerConnectionParms
7. İstemci kanal tanımlama çizelgesi (CCDT). CCDT ' nin yeri .NET uygulaması yapılandırma dosyasında belirtilir (yalnızca yönetilen bağlantılar için geçerlidir)
8. İstemci kanal tanımlama çizelgesi (CCDT). CCDT ' nin yeri, *MQCHLIB* ve *MQCHLTAB*ortam değişkenleri kullanılarak belirtilir.
9. İstemci kanal tanımlama çizelgesi (CCDT). CCDT ' nin yeri, istemci yapılanış kütüğü kullanılarak belirtilir

1-3 numaralı öğeler için, kanal tanımlaması, uygulama tarafından sağlanan değerlerden alan tarafından alan temelinde oluşturulur. Bu değerler farklı arabirimler kullanılarak sağlanabilir ve her biri için birden çok değer bulunabilir. Alan değerleri, verilen öncelik sırasının ardından kanal tanımına eklenir:

1. MQQueueManager oluşturucuda *connName* değerinin değeri
2. MQQueueManager.properties HASH çizelgesinden özelliklerin değerleri
3. MQEnvironment.properties HASH çizelgesinden özellik değerleri
4. Değerler MQEnvironment alanları olarak ayarlandı (örneğin, MQEnvironment.Hostname, MQEnvironment.Port)

4-6 numaralı öğeler için, tüm kanal tanımlaması değer olarak sağlanır. Kanal tanımlamasındaki belirlenmemiş alanlar sistem varsayılanlarını alır. Diğer tanımlama yöntemlerinin ve alanlarının diğer yöntemlerinden herhangi bir değer bu belirtimlerle birleştirilir.

7-9 numaralı öğeler için, tüm kanal tanımlaması CCDT ' den alınır. Kanal tanımlandığında belirtik olarak belirlenmeyen alanlar, sistem varsayılan değerlerini alır. Diğer tanımlama yöntemlerinin ve alanlarının diğer yöntemlerinden herhangi bir değer bu belirtimlerle birleştirilir.

Using channel exits in IBM WebSphere MQ .NET

İstemci bağ tanımlarını kullanıyorsanız, diğer istemci bağlantılarında olduğu gibi kanal çıkışlarını da kullanabilirsiniz. Yönetilen bağ tanımlarını kullanırsanız, uygun bir arabirimi gerçekleştiren bir çıkış programı yazmanız gerekir.

İstemci bağ tanımları

İstemci bağ tanımlarını kullanıyorsanız, kanal çıkışlarını Kanal çıkışlarındaaçıklandığı gibi kullanabilirsiniz. Yönetilen bağ tanımları için yazılan kanal çıkışlarını kullanamazsınız.

Yönetilen bağ tanımları

Yönetilen bir bağlantı kullanırsanız, bir çıkışı gerçekleştirmek için uygun arabirimi gerçekleştiren yeni bir .NET sınıfı tanımlamanız gerekir. WebSphere MQ paketinde üç çıkış arabirimi tanımlanir:

- MQSendExit
- MQReceiveExit
- MQSecurityExit

Not: Bu arabirimler kullanılarak yazılan kullanıcı çıkışları, yönetilmeyen ortamdaki kanal çıkışları olarak desteklenmez.

Aşağıdaki örnekte, üçünü gerçekleştiren bir sınıf tanımlanmaktadır:

```
class MyMQExits : MQSendExit, MQReceiveExit, MQSecurityExit
{
   // This method comes from the send exit
  byte[] SendExit(MQChannelExit       channelExitParms,
                  MQChannelDefinition channelDefinition,
                  byte[]              dataBuffer
                  ref int             dataOffset
                  ref int             dataLength
                  ref int             dataMaxLength)
  {
     // complete the body of the send exit here
  }


  // This method comes from the receive exit
  byte[] ReceiveExit(MQChannelExit       channelExitParms,
                     MQChannelDefinition channelDefinition,
                     byte[]              dataBuffer
                     ref int             dataOffset
                     ref int             dataLength
                     ref int             dataMaxLength)
  {
     // complete the body of the receive exit here
  }


  // This method comes from the security exit
  byte[] SecurityExit(MQChannelExit       channelExitParms,
                      MQChannelDefinition channelDefParms,
                      byte[]              dataBuffer
                      ref int             dataOffset
                      ref int             dataLength
                      ref int             dataMaxLength)
  {
     // complete the body of the security exit here
  }

}
```

Her çıkışa bir MQChannelExit ve bir MQChannelDefinition nesne eşgörünümü geçirilir. Bu nesneler, yordamsal arabirimde tanımlanan MQCXP ve MQCD yapılarını gösterir.

Bir gönderme çıkışı tarafından gönderilecek veriler ve bir güvenlik ya da alma çıkışta alınan veriler, çıkışa ilişkin parametreleri kullanarak belirtilir.

On entry, the data at offset *dataOffset* with length *dataLength* in the byte array *dataBuffer* is the data that is about to be sent by a send exit, and the data received in a security or receive exit. *dataMaxLength* değiştirgesi, *dataBuffer*içindeki çıkışa kadar uzunluk üst sınırını (*dataOffset*içinden) verir. Not: Bir güvenlik çıkışı için, çıkışa ilk kez çağrılırsa ya da iş ortağı sonu veri göndermek üzere seçildiyse, dataBuffer için boş değer (null) olabilir.

Dönüş sırasında, *dataOffset* ve *dataLength* değeri, .NET sınıflarının daha sonra kullanması gereken, döndürülen bayt dizisi içindeki göreli konum ve uzunluk değerine işaret edecek şekilde ayarlanmalıdır. Gönderme çıkışı için bu, göndermesi gereken verileri gösterir ve bir güvenlik ya da alma çıkışı için, yorumlanacak veriler belirtilir. Çıkış, olağan durumda bir bayt dizisi döndürmelidir; kural dışı durumlar, veri göndermeyi seçebilecek bir güvenlik çıkışıdır ve INIT ya da TERM nedenleriyle çağrılan herhangi bir çıkıştan çıkılır. Bu nedenle yazılabilecek en basit çıkış biçimi dataBuffer' ı döndürmekten başka bir şey yapmamaktadır.

Olabilecek en basit çıkış gövdesi şunlardır:

```
{
  return dataBuffer;
}
```

MQChannelDefinition sınıfı

V 7.5.0.6 Version 7.5.0, Fix Pack 6olanağından, yönetilen .NET istemci uygulaması ile belirtilen kullanıcı kimliği ve parola, istemci güvenlik çıkışa geçirilen IBM WebSphere MQ .NET MQChannelDefinition sınıfında ayarlanır. Güvenlik çıkışı, kullanıcı kimliğini ve parolayı MQCD.RemoteUserIdentifier ve MQCD.RemotePassword alanları (bkz. “Güvenlik çıkışı yazılıyor” sayfa 388).

Kanal çıkışlarının belirtilmesi (yönetilen istemci)

MQQueueManager nesnesini yaratırken (MQEnvironment ya da MQQueueManager oluşturucuda) bir kanal adı ve bağlantı adı belirtirseniz, kanal çıkışlarını iki şekilde belirleyebilirsiniz.

Öncelik sırasıyla şunlar olur:

1. MQQueueManager oluşturucuda MQC.SECURITY_EXIT_PROPERTY, MQC.SEND_EXIT_PROPERTY ya da MQC.RECEIVE_EXIT_PROPERTY gruplama etiketleri (hashtable) özellikleri geçirilebilir.
2. MQEnvironment SecurityExit, SendExit ya da ReceiveExit özellikleri ayarlanıyor.

Bir kanal adı ve bağlantı adı belirtmezseniz, kanal tanımından kullanılacak kanal, istemci kanalı tanımlama çizelgesinden (CCDT) alınan kanal tanımlamalarından çıkar. Kanal tanımlamasında saklanan değerleri geçersiz kılmanız mümkün değildir. Kanal tanımlama çizelgelerine ilişkin ek bilgi için İstemci kanal tanımlama çizelgesi ve “.NET içeren bir istemci kanal tanımlama çizelgesi kullanılması” sayfa 562 başlıklı konuya bakın.

Her durumda, belirtim aşağıdaki biçimi kullanarak bir dizginin biçimini alır:

```
Assembly_name(Class_name)
```

Class_name , IBM.WMQ.MQSecurityExit, IBM.WMQ.MQSendExit ya da IBM.WMQ.MQReceiveExit arabirimi (uygun olduğu gibi). *Assembly_name* , sınıfı barındıran düzeneğin dosya uzantısı da içinde olmak üzere tam olarak nitelenmiş yeridir. MQEnvironment ya da MQQueueManagerözelliklerini kullanıyorsanız, dizilimin uzunluğu 999 karakterle sınırlıdır. Ancak, CCDT ' de kanal çıkış adı belirtildiyse, bu değer 128 karakterle sınırlıdır. Gerekli olduğunda, .NET istemci kodu, dizgi belirtimini ayrıştırırken belirtilen sınıfın bir eşgörünümünü yükler ve yaratır.

Kanal çıkışı kullanıcı verilerinin belirtilmesi (yönetilen istemci)

Kanal çıkışları, bunlarla ilişkilendirilmiş kullanıcı verilerine sahip olabilir. MQQueueManager nesnenizi yaratırken (MQEnvironment ya da MQQueueManager oluşturucuda) bir kanal adı ve bağlantı adı belirtirseniz, kullanıcı verilerini iki şekilde belirtebilirsiniz.

Öncelik sırasıyla şunlar olur:

1. MQQueueManager oluşturucuda MQC.SECURITY_USERDATA_PROPERTY, MQC.SEND_USERDATA_PROPERTY ya da MQC.RECEIVE_USERDATA_PROPERTY gruplama etiketleri (hashtable) özellikleri geçirilebilir.
2. MQEnvironment SecurityUserVerileri, SendUserVerileri ya da ReceiveUserVeri özellikleri ayarlanıyor.

Bir kanal adı ve bağlantı adı belirtmezseniz, kullanılacak çıkış kullanıcı veri değerleri, istemci kanal tanımlama çizelgesinden (CCDT) alınan kanal tanımlamasından gelir. Kanal tanımlamasında saklanan değerleri geçersiz kılmanız mümkün değildir. Kanal tanımlama çizelgelerine ilişkin ek bilgi için İstemci kanal tanımlama çizelgesi ve “.NET içeren bir istemci kanal tanımlama çizelgesi kullanılması” sayfa 562 başlıklı konuya bakın.

Her durumda, belirtim 32 karakterle sınırlı olan bir dizgidir.

.NET ' te otomatik istemci yeniden bağlantısı

Beklenmeyen bir bağlantı sonu sırasında istemcinizin otomatik olarak bir kuyruk yöneticisine yeniden bağlanmasını sağlamanız gerekir.

Örneğin, kuyruk yöneticisi durdurursa ya da ağ ya da sunucu başarısız olursa, istemci beklenmedik bir şekilde kuyruk yöneticisinden bağlantısı kesilebilir.

Otomatik istemci yeniden bağlantısı olmadan, bağlantı başarısız olduğunda hata ortaya çıktı. Bağlantıyı yeniden kurmanıza yardımcı olması için hata kodunu kullanabilirsiniz.

Otomatik istemci yeniden bağlantı olanağını kullanan bir istemciye, yeniden bağlanabilir istemci adı verilir. Yeniden bağlanabilir bir istemci yaratmak için, kuyruk yöneticisine bağlanırken yeniden bağlanma seçenekleri çağrılan belirli seçenekleri belirtin.

İstemci uygulaması bir WebSphere MQ .NET istemciyse, kuyruk yöneticisi yaratmak için `MQQueueManager` sınıfını kullandığınızda, CONNECT_OPTIONS_PROPERTY için uygun bir değer belirterek, otomatik istemci yeniden bağlantı almayı tercih edebilir. CONNECT_OPTIONS_PROPERTY değerlerinin ayrıntıları için Yeniden yapılandırma seçenekleri başlıklı konuya bakın.

İstemci uygulamasının her zaman aynı adı taşıyan bir kuyruk yöneticisine, aynı kuyruk yöneticisine ya da istemci bağlantı çizelgesinde aynı `QMNAME` ile tanımlanmış kuyruk yöneticilerine bağlanıp bağlanmayacağını seçebilirsiniz (ayrıntılar için CCDT ' de Kuyruk Yöneticisi Grupları ' a bakın).

Güvenli Yuva Arabirimi Katmanı (SSL) desteği

Aşağıdaki bölüm yönetilen istemci için geçerli değildir.

.NET istemci uygulamaları için WebSphere MQ sınıfları SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) şifrelemesini destekler. SSL, iletişim şifrelemesi, kimlik doğrulaması ve ileti bütünlüğü sağlar. Bu, genellikle İnternet üzerinde ya da bir intranet içinde herhangi iki eş arasında iletişim sağlamak için kullanılır.

SSL olanağının etkinleştirilmesi

SSL yalnızca istemci bağlantıları için desteklenir. SSL ' yi etkinleştirmek için, kuyruk yöneticisiyle iletişim kurarken kullanılacak CipherSpec değerini belirtmeniz gerekir; bu değer, hedef kanaldaki CipherSpec kümesiyle eşleşmelidir.

SSL ' yi etkinleştirmek için, MQEnvironment 'ın SSLCipherSpec durağan üye değişkenini kullanarak CipherSpec değerini belirtin. Aşağıdaki örnek, NULL_MD5:' un CipherSpec ile SSL gerektirecek şekilde ayarlanmış olan SECURE.SVRCONN.CHANNELadlı bir SVRCONN kanalına bağlanır.

```
MQEnvironment.Hostname         = "your_hostname";
MQEnvironment.Channel          = "SECURE.SVRCONN.CHANNEL";
MQEnvironment.SSLCipherSpec    = "NULL_MD5";
MQEnvironment.SSLKeyRepository = "C:\mqm\key";
MQQueueManager qmgr = new MQQueueManager("your_Q_manager");
```

CipherSpecslistesi için bkz. Specifying CipherSpecs .

The SSLCipherSpec property can also be set using the MQC.SSL_CIPHER_SPEC_PROPERTY in the hash table of connection properties.

SSL kullanarak başarıyla bağlanmak için, istemci anahtar deposunun, kuyruk yöneticisi tarafından sunulan sertifikanın doğrulanabileceği Sertifika Yetkilisi kök sertifikaları zinciriyle birlikte ayarlanması

gerekir. Benzer şekilde, SVRCONN kanalında SSLClientAuth MQSSL_CLIENT_AUTH_REQUIRED olarak ayarlandıysa, istemci anahtar deposu, kuyruk yöneticisi tarafından güvenilen bir tanıtıcı özel sertifika içermeli.

Kuyruk yöneticisinin Ayırt Edici Adı 'nı kullanma

Kuyruk yöneticisi, kendisini *Ayırt Edici Ad* (DN) içeren bir SSL sertifikası kullanarak tanımlar.

Bir WebSphere MQ .NET istemci uygulaması, doğru kuyruk yöneticisiyle iletişim kurduğundan emin olmak için bu DN ' yi kullanabilir. MQEnvironment 'ın sslPeerAd değişkeni kullanılarak bir DN örünü belirtildi. Örneğin:

```
MQEnvironment.SSLPeerName = "CN=QMGR.*, OU=IBM, OU=WEBSPHERE";
```

Ancak kuyruk yöneticisi, QMGR. ' un başında bir Common Name değeri olan bir sertifika sunarsa, bağlantının başarılı olmasına izin verir. ve en az iki Kuruluş Birimi adı, ilkinin IBM ve ikinci WEBSPHERE olması gerekir.

The SSLPeerName property can also be set using the MQC.SSL_PEER_NAME_PROPERTY in the hash table of connection properties. Eş adları ayarlamaya ilişkin ayırt edici adlar ve kurallara ilişkin ek bilgi edinmek için bkz. Durumu.

SSLPeerName ayarlandıysa, bağlantılar yalnızca geçerli bir örüntü olarak ayarlandıysa ve kuyruk yöneticisi eşleşen bir sertifika sunarsa başarılı olur.

SSL kullanılırken hata işlenirken hata oluştu

Aşağıdaki neden kodları, SSL kullanan bir kuyruk yöneticisine bağlanırken .NET için WebSphere MQ sınıfları tarafından yayınlanabilir:

MQRC_SSL_NOT_ALLOWED
SSLCipherSpec özelliği ayarlıydı, ancak bağ tanımları bağlantısı kullanıldı. SSL ' yi yalnızca istemci bağlantısı destekler.

MQRC_SSL_PEER_NAME_MISSMATCH
SSLPeerName özelliğinde belirtilen DN kalıbı, kuyruk yöneticisi tarafından sunulan DN ile eşleşmedi.

MQRC_SSL_PEER_NAME_ERROR
SSLPeerName özelliğinde belirtilen ayırt edici ad (DN) kalıbı geçerli değil.

.NET İzleme Programının Kullanılması

Önemli bilgi için Yalnızca Windows 'ta birincil kuruluşla kullanılabilen özellikler konusuna bakın.

.NET Monitor, WebSphere MQ tetikleyicisi izleyicisine benzer bir uygulamadır. İzlenen bir kuyruğun üzerinde bir ileti alındığında somutlaştırılan .NET bileşenleri yaratabilir ve bu iletiyi işleyebilirsiniz. .NET Monitor, `runmqdnm` komutu tarafından başlatılır ve `endmqdnm` komutu tarafından durdurulur. Bu komutlara ilişkin ayrıntılar için bkz. runmqdnm ve endmqdnm.

.NET Monitor 'u kullanmak için, amqmdnm.dlliçinde tanımlanmış olan IMQObjectTrigger arabirimini gerçekleştiren bir bileşen yazıyorsunuz.

Bileşenler işlemsel ya da işlemsel olmayan bir bileşen olabilir. Bir işlemsel bileşen System.EnterpriseServices.ServicedComponent ' den edinilmelidir ve RequiresTransaction ya da SupportsTransactionolarak kaydedilmelidir. .NET Monitor zaten bir işlem başlatmış olduğundan, bu değer RequiresNew olarak kaydedilmemelidir.

Bileşen, `runmqdnm`içinden MQQueueManager, MQQueue ve MQMessage nesnelerini alır. Ayrıca, `runmqdnm` başlatıldığında, *-u* komut satırı seçeneği kullanılarak bir Kullanıcı Parametresi dizgisi de alabilir. Bileşeninizin, bir MQMessage nesnesindeki izlenen kuyruğa gelen bir iletinin içeriğini aldığını unutmayın. Kuyruk yöneticisine bağlanmak, kuyruğu açmak ya da iletinin kendisini almak zorunda değildir. Daha sonra, bu bileşenin iletiyi uygun olarak işlemesi ve .NET Monitor 'a geri dönüş denetimi olması gerekir.

If your component has been written as a transactional component, it registers to commit or roll back the transaction using the facilities provided by System.EnterpriseServices.ServicedComponent.

Bileşen MQQueueManager ve MQQueue nesnelerinin yanı sıra iletiyi aldıktan sonra, bu ileti için bağlam bilgilerini tamamlar ve örneğin, WebSphere MQ' a ayrı ayrı bağlanmak gerekmeden aynı kuyruk yöneticisine başka bir kuyruk açabilir.

Örnek kod parçaları

Bu konuda, .NET Monitor 'dan ileti alan ve bu iletiyi yazdırmak için kullanılan bileşenlere ilişkin iki örnek yer alır; bunlardan biri işlemsel işlemeyi ve diğeri hareketsel olmayan işlemeyi kullanır. Üçüncü bir örnek, her iki örnek için de geçerli olan ortak yardımcı program yordamlarını gösterir. Tüm örnekler C# ' de.

Örnek 1: İşlemsel işleme

```
/********************************************************************/
/* Licensed materials, property of IBM                              */
/* 63H9336                                                          */
/* (C) Copyright IBM Corp. 2005, 2024.                              */
/********************************************************************/
using System;
using System.EnterpriseServices;

using IBM.WMQ;
using IBM.WMQMonitor;

[assembly: ApplicationName("dnmsamp")]

// build:
//
// csc -target:library -reference:amqmdnet.dll;amqmdnm.dll TranAssembly.cs
//
// run (with dotnet monitor)
//
// runmqdnm -m <QMNAME> -q <QNAME> -a dnmsamp.dll -c Tran

namespace dnmsamp
{
  [TransactionAttribute(TransactionOption.Required)]
  public class Tran : ServicedComponent, IMQObjectTrigger
  {
    Util util = null;

    [AutoComplete(true)]
    public void Execute(MQQueueManager qmgr, MQQueue queue,
        MQMessage message, string param)
    {
      util = new Util("Tran");

      if (param != null)
        util.Print("PARAM: '" +param.ToString() + "'");

      util.PrintMessage(message);

      //System.Console.WriteLine("SETTING ABORT");
      //ContextUtil.MyTransactionVote = TransactionVote.Abort;

      System.Console.WriteLine("SETTING COMMIT");
      ContextUtil.SetComplete();
      //ContextUtil.MyTransactionVote = TransactionVote.Commit;
    }
  }
}
```

Örnek 2: Hareket dışı işleme

```
/********************************************************************/
/* Licensed materials, property of IBM                              */
/* 63H9336                                                          */
/* (C) Copyright IBM Corp. 2005, 2024.                              */
/********************************************************************/

using System;
```

```
using IBM.WMQ;
using IBM.WMQMonitor;

// build:
//
// csc -target:library -reference:amqmdnet.dll;amqmdnm.dll NonTranAssembly.cs
//
// run (with dotnet monitor)
//
// runmqdnm -m <QMNAME> -q <QNAME> -a dnmsamp.dll -c NonTran
namespace dnmsamp
{
  public class NonTran : IMQObjectTrigger
  {
    Util util = null;

    public void Execute(MQQueueManager qmgr, MQQueue queue,
        MQMessage message, string param)
    {
      util = new Util("NonTran");

      try
      {
        util.PrintMessage(message);
      }

      catch (Exception ex)
      {
        System.Console.WriteLine(">>> NonTran\n{0}", ex.ToString());
      }
    }
  }
}
```

Örnek 3: Ortak yordamlar

```
/*********************************************************************/
/* Licensed materials, property of IBM                               */
/* 63H9336                                                           */
/* (C) Copyright IBM Corp. 2005, 2024.                               */
/*********************************************************************/

using System;

using IBM.WMQ;

namespace dnmsamp
{
    /// <summary>
    /// Summary description for Util.
    /// </summary>
    public class Util
    {
    /* ---------------------------------------------------------------- */
    /* Default prefix string of the namespace.                           */
    /* ---------------------------------------------------------------- */
    private string prefixText = "dnmsamp";

    /* ---------------------------------------------------------------- */
    /* Constructor that takes the replacement prefix string to use.     */
    /* ---------------------------------------------------------------- */
    public Util(String text)
    {
      prefixText = text;
    }

    /* ---------------------------------------------------------------- */
    /* Display an arbitrary string to the console.                      */
    /* ---------------------------------------------------------------- */
    public void Print(String text)
    {
      System.Console.WriteLine("{0} {1}\n", prefixText, text);
    }
```

```
 /* ------------------------------------------------------------------ */
    /* Display the content of the message passed to the console.          */
    /* ------------------------------------------------------------------ */
    public void PrintMessage(MQMessage message)
    {
      if (message.Format.CompareTo(MQC.MQFMT_STRING) == 0)
      {
        try
        {
          string messageText = message.ReadString(message.MessageLength);

          Print(messageText);
        }

        catch(Exception ex)
        {
          Print(ex.ToString());
        }
      }
      else
      {
        Print("UNRECOGNISED FORMAT");
      }
    }

    /* ------------------------------------------------------------------ */
    /* Convert the byte array into a hex string.                          */
    /* ------------------------------------------------------------------ */
    static public string ToHexString(byte[] byteArray)
    {
      string hex = "0123456789ABCDEF";

      string retString = "";

      for(int i = 0; i < byteArray.Length; i++)
      {
        int h = (byteArray[i] & 0xF0)>>4;
        int l = (byteArray[i] & 0x0F);

        retString += hex.Substring(h,1) + hex.Substring(l,1);
      }

      return retString;
    }

    }
}
```

WebSphere MQ .NET programlarının derlenmesi

Çeşitli dillerde yazılmış .NET uygulamalarını derlemek için kullanılan örnek komutları.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

.NET için WebSphere MQ sınıflarını kullanarak C# uygulaması oluşturmak için aşağıdaki komutu kullanın:

```
csc /t:exe /r:System.dll /r:amqmdnet.dll /lib:MQ_INSTALLATION_PATH\bin /out:MyProg.exe MyProg.cs
```

.NET için WebSphere MQ sınıflarını kullanarak bir Visual Basic uygulaması oluşturmak için aşağıdaki komutu kullanın:

```
vbc /r:System.dll /r:MQ_INSTALLATION_PATH\bin\amqmdnet.dll /out:MyProg.exe MyProg.vb
```

.NET için WebSphere MQ sınıflarını kullanarak bir yönetilen C++ uygulaması oluşturmak için aşağıdaki komutu kullanın:

```
cl /clr MQ_INSTALLATION_PATH\bin Myprog.cpp
```

Diğer diller için, dil sağlayıcısının sağladığı belgelere bakın.

WebSphere MQ .NET programlarının izlenmesi

WebSphere MQ .NET ' te, izleme olanağını, MQI kullanarak WebSphere MQ programlarında olduğu gibi başlatır ve denetleyebilirsiniz.

Ancak, süreç ve iş parçacığı tanıtıcılarını ve adlandırılmış süreçleri belirtmenize olanak tanıyan strmqtrc komutunun -i ve -p parametreleri hiçbir etkisizmez.

Normalde izleme olanağını yalnızca IBM hizmetinin isteği üzerine kullanmanız gerekir.

İzleme komutlarına ilişkin bilgi için bkz. Using trace on Pencereler .

Microsoft Windows Communication Foundation (WCF) içinIBM WebSphere MQ özel kanalı

The Microsoft Windows Communication Foundation (WCF) custom channel for IBM WebSphere MQ sends and receives messages between WCF clients and services.

İlgili kavramlar

“.NET 3 ile WCF için WebSphere MQ özel kanalının kullanımına giriş” sayfa 571
Overview of the information available for programmers using the WebSphere MQ custom channel for Pencereler Communication Foundation (WCF) with .NET 3.

“WCF için WebSphere MQ özel kanallarını kullanma” sayfa 575
Pencereler Communication Foundation (WCF) için WebSphere MQ V7 özel kanallarını kullanan programcılar için kullanıma sunulan bilgilere genel bakış.

“WCF örneklerinin kullanılması” sayfa 591
Windows Communication Foundation (WCF) örnekleri, WebSphere MQ özel kanalının nasıl kullanılabileceğiyle ilgili bazı basit örnekler sağlar.

“ WebSphere MQiçin WCF özel kanalındaki sorun belirleme” sayfa 597
WebSphere MQ kodunun çeşitli kısımlarıyla ilgili ayrıntılı bilgileri toplamak için WebSphere MQ izlemesini kullanabilirsiniz. Pencereler Communication Foundation (WCF) kullanılırken, WCF özel kanal izlemesi için Microsoft WCF altyapı izlemesiyle bütünleştirilmiş ayrı bir izleme çıkışı yaratılır.

.NET 3 ile WCF için WebSphere MQ özel kanalının kullanımına giriş

Overview of the information available for programmers using the WebSphere MQ custom channel for Pencereler Communication Foundation (WCF) with .NET 3.

WCF için WebSphere MQ özel kanalı nedir?

WebSphere MQ için özel kanal, Microsoft Windows Communication Foundation (WCF) birleşik programlama modelini kullanan bir iletim kanalı.

Microsoft .NET 3 'te tanıtılan Microsoft Windows Communication Foundation çerçevesi, .NET uygulamalarının ve hizmetlerinin, bunları bağlamak için kullanılan iletim ve iletişim kurallarından bağımsız olarak geliştirilmesini sağlayarak, hizmetin ya da uygulamanın konuşlandırıldığı ortama göre kullanılacak alternatif aktarımlardan ya da yapılandırmalardan bağımsız olarak geliştirilmesini sağlar.

Bağlantılar, gereken bileşimi içeren bir kanal yığını oluşturarak WCF tarafından çalıştırma zamanında yönetilir:

- İletişim kuralı öğeleri: WS-* standartları gibi destek protokollerine hiçbirinin, bir ya da daha fazlasının eklenebileceği, isteğe bağlı bir öğe kümesi.
- İleti kodlayıcısı: İletinin aktarım kanalı biçimine diziselleştirmesini denetleyen yığındaki zorunlu bir öğe.
- İletim kanalı: Serileştirilmiş iletiyi uç noktasına taşımaktan sorumlu yığın içinde zorunlu bir öğe.

WebSphere MQ için özel kanal bir iletim kanalıdır ve WCF özel bağlaması kullanılarak uygulamanın gerektirdiği bir ileti kodlayıcı ve isteğe bağlı protokolle eşlenmelidir. In this way, applications which have been developed to use WCF can use the custom channel for WebSphere MQ to send and receive data in the same way as they use the built-in transports provided by Microsoft, enabling simple integration with

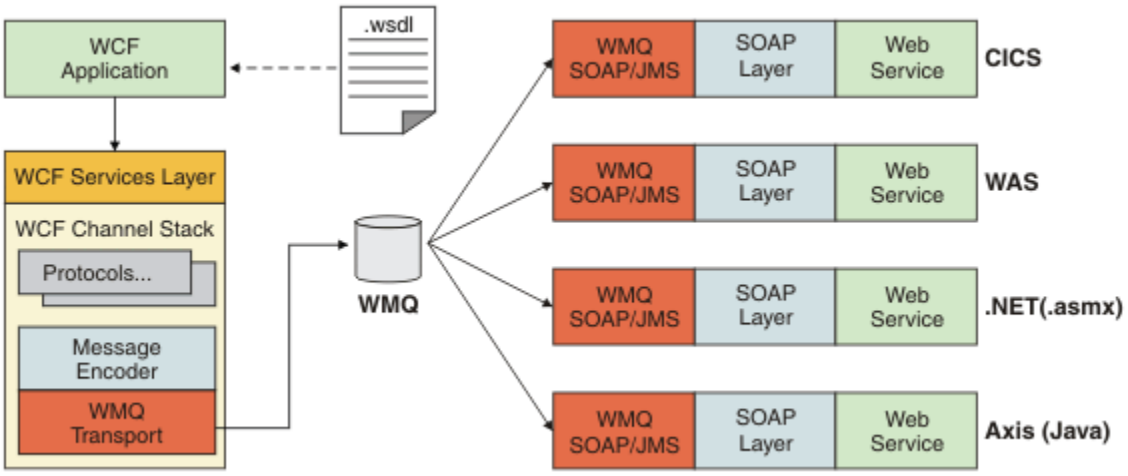
the asynchronous, scalable, and reliable messaging functions of WebSphere MQ. Desteklenen işlevlerin tam listesi için bkz. “WCF Özel kanal özellikleri ve yetenekleri” sayfa 575.

WCF için WebSphere MQ özel kanalını ne zaman ve neden kullanırım?

The WebSphere MQ custom channel can be used to send and receive messages between WCF clients and services in the same way as the built-in transports provided by Microsoft, enabling applications to access the features of WebSphere MQ within the WCF unified programming model.

A typical usage pattern scenario of the WebSphere MQ custom channel for WCF is as an interface to web services hosted over WebSphere MQ (SOAP/JMS)

Messages are carried using the SOAP over JMS message format of WebSphere MQ, enabling WCF clients and services to also call or be called by other WebSphere MQ applications or hosting environments which are compatible with this format, including web services and clients running in WebSphere Application Server, CICS, Axis v1 (Java), and .asmx (.NET), as shown in the following diagram:



JMS üzerinden SOAP ile ilgili ayrıntılar için bkz. “SOAP içinWebSphere MQ iletimi” sayfa 908

An example of a typical scenario from the diagram would be:

1. A web Service hosted within WebSphere Application Server and exposed over WebSphere MQ using the support for SOAP over JMS within WebSphere Application Server
2. Hizmeti tanımlayan WSDL belgesi, daha sonra özel kanal da dahil olmak üzere uygun bir WCF kanal yığını yaratacak bir istemci yetkili sunucusu ve yapılanış oluşturmak için WCF aracı tarafından kullanılabilir.
3. Daha sonra, istemci uygulaması, web hizmetini diğer herhangi bir web hizmeti ile aynı şekilde başlatmak için yetkili sunucuyu kullanabilir.

Kanal genellikle bir WCF text/SOAP iletisi kodlayıcısıyla birlikte kullanılır; ancak, gerekirse kanal, diğer WCF ileti kodlayıcılarıyla eşlenmiş olabilir. Alternatif kodlayıcılar kullanılarak, JMS üzerinde SOAP ' ı desteklemeyen yerel WebSphere MQ uygulamalarıyla sınırlı bütünleştirme de sağlanabilir, ancak bu, kanalın birincil rolü değildir.

Bir WCF ortamında özel kanal kullanılmasının temel yararları şunlardır:

- Zamanuyumsuz çağırma: Yanıtlar ve çoklu sekme gibi özelliklerin yeniden yönlendirmesi gibi hizmetin ve özelliklerin kullanılabilirliğinden istemcinin ayrıldığı müşteri operasyonlarını destekler ve unutur.
- Güvenilir ölçekleme özellikleri: Kuyruk tabanlı ileti sistemi, bir sisteme tahmin edilebilir bir şekilde kapasite eklenmesine olanak sağlar.
- Hizmet kalitesi: İletiler elle tutulur ve izlenebilir, kolaylıkla yönetilebilir ve yönetilebilir.

WCF için WebSphere MQ özel kanalına ilişkin yazılım gereksinimleri ve kuruluş yönergeleri

Bu konuda, WCF için WebSphere MQ özel kanalına ilişkin yazılım gereksinimleri ve kuruluş bilgileri özetlenmiştir.

WCF için WebSphere MQ özel kanalı, yalnızca WebSphere MQ V7 ya da daha yüksek kuyruk yöneticilerine bağlanabilir.

WebSphere MQiçin WCF özel kanalına ilişkin yazılım gereksinimleri

This information lists the software requirements for the WCF custom channel for WebSphere MQ.

Runtime Environment

- Microsoft .NET Framework v3.0 or higher must be installed on the host machine.
- *Java ve .NET Messaging ve Web Hizmetleri* is installed by default as part of the WebSphere MQ 7.0.1 installer. Özel kanal için gereken .NET yapıbirimlerini Genel Derleme Önbelleğiyle Kurar.

Not: WebSphere MQ V7.0.1kuruluşundan önce Microsoft .NET Framework v2.0 ya da üstü kurulu değilse, WebSphere MQ ürün kuruluşu hata vermeden devam eder, ancak WebSphere MQ özel kanalı kullanılamaz. If the .NET Framework is installed after installing WebSphere MQ 7.0.1, then the WebSphere MQ custom channel must be activated by running the *WMQInstallDir*\bin\amqiRegisterdotNet.cmd script, where *WMQInstallDir* is the directory where WebSphere MQ 7.0.1 is installed. Bu komut dosyası, gerekli düzenekleri Global Assembly Cache (GAC) içine kurar. Alınan işlemleri kaydeden bir amqi*.log dosyası kümesi, %TEMP% dizininde oluşturulur. .NET v3.0 düzeyine yükseltilirse ya da daha önceki bir sürümden (örneğin, .NET v2.0) daha yüksek bir sürüme yükseltilirse, amqiRegisterdotNet.cmd komut kütüğünü yeniden çalıştırmak gerekmez.

Geliştirme ortamı

- Microsoft Visual Studio 2008 ya da Windows Software Development Kit for .NET 3.0 ya da üstü.
- Örnek çözüm dosyalarını oluşturmak için anasistem makinesindeMicrosoft .NET Framework V3.5 ya da sonraki sürümü kurulu olmalıdır.

Not: WebSphere MQ V7.0.1kuruluşundan önce Microsoft .NET Framework v2.0 ya da üstü kurulu değilse, WebSphere MQ ürün kuruluşu hata vermeden devam eder, ancak WebSphere MQ özel kanalı kullanılamaz. If the .NET Framework is installed after installing WebSphere MQ 7.0.1, then the WebSphere MQ custom channel must be activated by running the *WMQInstallDir*\bin\amqiRegisterdotNet.cmd script, where *WMQInstallDir* is the directory where WebSphere MQ 7.0.1 is installed. Bu komut dosyası, gerekli düzenekleri Global Assembly Cache (GAC) içine kurar. Alınan işlemleri kaydeden bir amqi*.log dosyası kümesi, %TEMP% dizininde oluşturulur. .NET v3.0 düzeyine yükseltilirse ya da daha önceki bir sürümden (örneğin, .NET v2.0) daha yüksek bir sürüme yükseltilirse, amqiRegisterdotNet.cmd komut kütüğünü yeniden çalıştırmak gerekmez.

WebSphere MQ Custom Channel for WCF: What's installer?

WebSphere MQ için özel kanal, Microsoft Windows Communication Foundation (WCF) birleşik programlama modelini kullanan bir iletim kanalı. Özel kanal, varsayılan olarak WebSphere MQ 7.0.1 kuruluşunun bir parçası olarak kurulur.

WCF içinWebSphere MQ özel kanalı

The WebSphere MQ custom channel for WCF is installed by default as part of the WebSphere MQ 7.0.1 installation; The custom channel and its dependencies are contained within the Java and .NET Messaging and Web Services component, which is installed by default. Önceki bir sürümden WebSphere MQ 7.0.1 'e yükseltirken, Java and .NET Messaging and Web Services bileşeni daha önceki bir kuruluma önceden kurulduysa, güncelleme varsayılan olarak WebSphere MQ özel kanalı WCF' ye kurulur.

`Java and .NET Messaging and Web Services` bileşeni `IBM.XMS.WCF.dll` dosyasını içerir ve `IBM.XMS.WCF.dll` dosyası, WCF arabirim sınıflarını içeren ana özel kanal düzeneğidir. Bu dosya, Global Assembly Cache (GAC) içine kurulur ve şu dizinde de bulunur: *MQ_INSTALLATION_PATH*`\bin` burada *MQ_INSTALLATION_PATH* , WebSphere MQ 7.0.1 ürününün kurulu olduğu dizindir.

Özel kanalı kullanmak için gereken temel sınıflar *Ad Alanı: IBM.XMS.WCF* ve:

İletim Bağ Tanımı Adı	IBM.XMS.WCF.SoapJmsIbmTransportBindingElement
İletim Bağ Tanımı İçe Aktarıcısı	IBM.XMS.WCF.SoapJmsIbmTransportBindingElementImporter
İletim Bağ Tanımı Yapılanışı	IBM.XMS.WCF.SoapJmsIbmTransportBindingElementConfig

WebSphere MQ özel kanal örnekleri

Örnekler, WebSphere MQ özel kanalının WCF için nasıl kullanılabileceği ile ilgili bazı basit örnekler sağlar. Örnekler ve bunların ilişkili dosyaları, *MQ_INSTALLATION_PATH*`\tools\wcf\samples\` dizininde bulunur; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' nın kuruluş dizinidir. WebSphere MQ özel kanal örnekleriyle ilgili daha fazla bilgi için bkz. “WCF örneklerinin kullanılması” sayfa 591

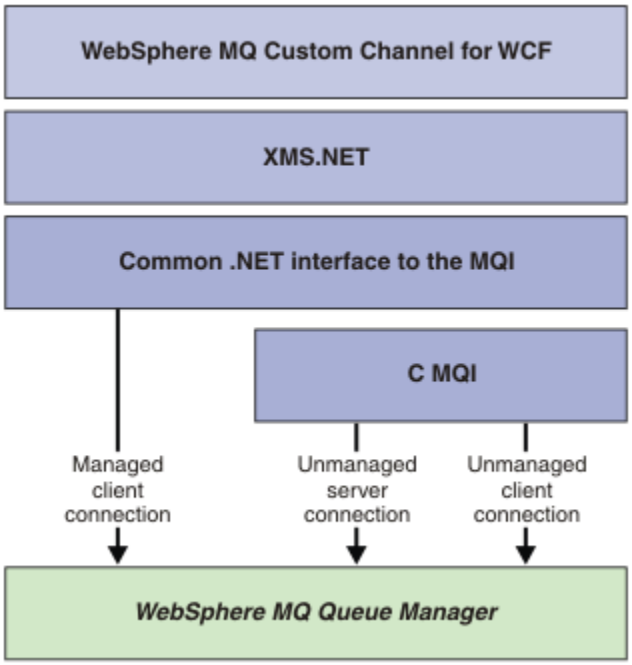
svcutil.exe.config

`svcutil.exe.config` , Microsoft WCF `svcutil` istemcisi yetkili sunucu oluşturma aracının özel kanalı tanımasını sağlamak için gereken yapılandırma ayarlarına bir örnektir. `svcutil.exe.config` dosyası, *MQ_INSTALLATION_PATH*`\tools\wcf\docs\examples\` dizininde bulunur; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' un kuruluş dizinidir. `svcutil.exe.config`kullanımıyla ilgili daha fazla bilgi için bkz. “Çalışan bir hizmetteki meta verileri içeren svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılandırma dosyaları oluşturulması” sayfa 588.

WCF mimarisi

WCF için WebSphere MQ özel kanalı, `IBM Message Service Client for .NET (XMS .NET)` API' nin üst kısmında tümleştirilmiştir.

WCF mimarisi aşağıdaki şemada gösterilmektedir:



Gerekli tüm bileşenler varsayılan olarak WebSphere MQ V7.0.1 kuruluşuyla kurulur.

Üç bağlantı şunlardır: Yönetilen istemci bağlantıları, Yönetilmeyen sunucu bağlantıları ve yönetilmeyen istemci bağlantıları. Bu bağlantılarla ilgili daha fazla bilgi için bkz. “WCF bağlantısı seçenekleri” sayfa 579.

WCF için WebSphere MQ özel kanallarını kullanma

Pencereler Communication Foundation (WCF) için WebSphere MQ V7 özel kanallarını kullanan programcılar için kullanıma sunulan bilgilere genel bakış.

Microsoft Pencereler Communication Foundation, Microsoft .NET Framework 3 içindeki web hizmetlerini ve ileti sistemi desteğini alt üst eder. WebSphere MQ V7 can now be used as a custom channel within WCF in the .NET Framework 3 in the same manner as the built-in channels offered by Microsoft.

Özel kanalda taşınan iletiler, WebSphere MQ V7'nin JMS uygulaması üzerinden SOAP' a göre biçimlendirilir. Applications can then communicate with services hosted by WCF or by the WebSphere SOAP over JMS service infrastructure. JMS üzerinden SOAP ile ilgili ayrıntılar için bkz. “SOAP içinWebSphere MQ iletimi” sayfa 908

WCF Özel kanal özellikleri ve yetenekleri

WCF özel kanal özellikleri ve yetenekleriyle ilgili bilgi için aşağıdaki konuları kullanın.

WCF özel kanal şekilleri

WebSphere MQ ' nun Microsoft Windows Communication Foundation (WCF) özel kanallarında olduğu gibi kullanılabileceği özel kanal şekillerine genel bakış.

WCF için WebSphere MQ özel kanalı iki kanal şeklini destekler:

- Tek Yönlü
- İstek-yanıt

WCF, barındırılmakta olan hizmet sözleşmesine göre otomatik olarak kanal şeklini seçer.

Yalnızca **IsOneWay** parametresini kullanan yöntemleri içeren sözleşmeler, tek yönlü kanal şekli tarafından bakıma alınmakta, örneğin:

```
    [OperationContract(IsOneWay = true)]
    void printString(String text);
```

Tek yönlü ve istek-yanıt yöntemlerinin bir karışımının ya da tüm istek-yanıt yöntemlerinin bir karışımı içeren sözleşmeler, istek yanıtlama kanalı şekli tarafından bakıma alınmadır. Örneğin:

```
    [OperationContract]
    int subtract(int a, int b);

    [OperationContract(IsOneWay = true)]
    void printString(string text);
```

Not: Tek yönlü ve istek yanıtlama yöntemlerini aynı sözleşmede birlikte kullanırken, özellikle tek yönlü yöntemler, hizmetten boş bir yanıt alınıncaya kadar beklediğinden, bu davranışın, özellikle de karma bir ortam içinde çalışırken kullanılması amaçlanan gibi olduğundan emin olmalısınız.

Tek yönlü kanal

WCF için WebSphere MQ tek yönlü özel kanal (örneğin, tek yönlü kanal şeklini kullanarak bir WCF istemcisinden ileti göndermek) kullanılır. Kanal, iletileri yalnızca tek bir yönde gönderebilir; örneğin, bir istemci kuyruk yöneticisinden bir WCF hizmetindeki bir kuyruğa gönderme yapabilir.

İstek-yanıt kanalı

WCF için WebSphere MQ istek yanıtlama özel kanalı kullanılır; örneğin, iletileri zamanuyumsuz olarak iki yönde göndermek için; zamanuyumsuz ileti sistemi için aynı istemci yönetim ortamının kullanılması gerekir. Kanal, iletileri tek bir yöne (örneğin, bir istemci kuyruk yöneticisinden bir WCF hizmetindeki bir kuyruğa gönderebilir) gönderebilir ve daha sonra, WCF ' den istemci kuyruk yöneticisinde bir kuyruğa bir yanıt iletisi gönderebilir.

WCF URI değiştirge adları ve değerleri

connectionFactory
connectionFactory parametresi gerekli. Bu parametrenin sözdizimi için bkz. Web hizmeti konuşlandırması için URI sözdizimi ve parametreleri .

initialContextFactory
initialContextFactory parametresi gerekli ve WebSphere Application Server ve diğer ürünlerle uyumluluk için "com.ibm.mq.jms.Nojndi" olarak ayarlanmalıdır (bkz. “WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması” sayfa 965).

WCF için özel kanal garantili teslim

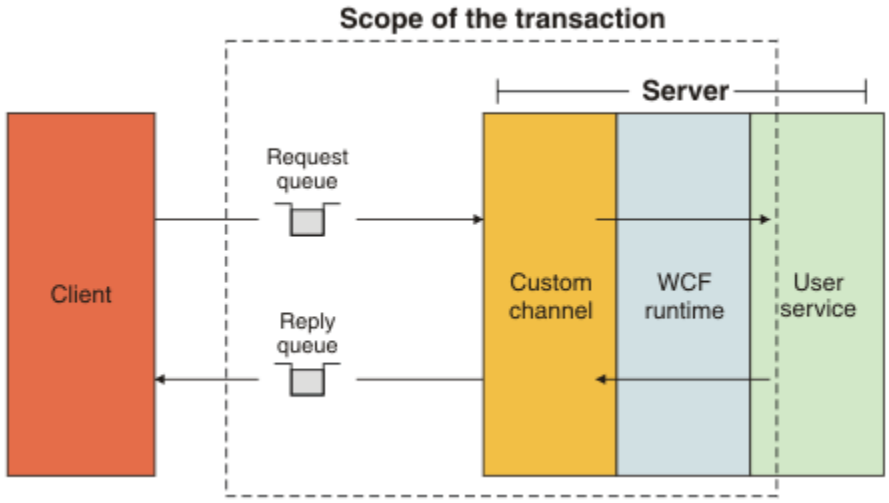
Bir hizmet isteğinin ya da yanıtının geçersiz olduğunu ve kaybedilmediğini garanti altına almak için Güvenli Teslim 'e garanti edilir.

Bir istek iletisi alındı ve herhangi bir yanıt iletisi, yürütme zamanı hatası durumunda geri döndürülebilecek bir yerel hareket eşitleme noktası altında gönderildi. Bu başarısızlıklara ilişkin örnekler: Hizmet tarafından yayınlanan işlenemeyen bir kural dışı durum, iletiyi hizmete gönderememek ya da yanıt iletisinin teslim edilmemesi.

`AssuredDelivery` , bir hizmet sözleşmesinde alınan istek iletilerinin ve bir hizmetten gönderilen herhangi bir yanıt iletisinin, çalıştırma zamanı hatası durumunda kaybedilmediğini garanti etmek için bir hizmet sözleşmesinde belirtilebilecek olan güvenli teslim özniteliğinden söz eder.

Sistem arızası ya da güç kesintisi durumunda iletilerin aynı zamanda korunmasını sağlamak için iletiler kalıcı olarak gönderilmelidir. Kalıcı iletileri kullanmak için, istemci uygulamasının uç noktası URI 'sında belirtilen bu seçeneğe sahip olması gerekir. URI özelliklerini ayarlama hakkında daha fazla bilgi için bakınız: URI sözdizimi ve Web hizmeti konuşlandırması için parametreler.

Dağıtımlı hareketler desteklenmez ve hareketin kapsamı, WebSphere MQtarafından gerçekleştirilen istek ve yanıt iletisi işlemenin ötesine geçmiyor. Hizmet içinde gerçekleştirilen tüm işler, iletinin yeniden alınmasına neden olan bir hatanın sonucu olarak yeniden çalıştırılabilir. Aşağıdaki çizge işlemin kapsamını gösterir:



Assured delivery is enabled by applying the `AssuredDelivery` attribute to the service class as shown in the following example:

```
[AssuredDelivery]
class TestCalculatorService : IWMQSampleCalculatorContract
{
    public int add(int a, int b)
    {
        int ans = a + b;
        return ans;
    }
}
```

`AssuredDelivery` özniteliğini kullanırken, aşağıdaki noktalardan haberdar olmalısınız:

- Bir kanal, bir iletinin geriye işlenip geri gönderilip alınmadığını saptadığında, ileti bir zehir iletisi olarak kabul edilir ve yeniden işlenmek üzere istek kuyruğuna döndürülmez. Örneğin, alınan ileti doğru biçimlendirilmediyse ya da bir hizmete dağıtılamazsa. Bir hizmet işleminden atılan işlenmeyen kural dışı durumlar, ileti, istek kuyruğunun geriletme eşiği özelliği tarafından belirtilen süre üst sınırını yeniden teslim edilinceye kadar her zaman yeniden içerilir. Daha fazla bilgi için bkz. “WCF özel kanal zehirli iletileri” sayfa 578
- Kanal, hareket bütünlüğünü zorlamak için tek bir yürütme parçası kullanarak, her istek iletisini bir atomik işlem olarak okuma, işleme ve yanıtlama işlemini gerçekleştirir. Hizmet işlemlerinin koşutzamanlı olarak çalışmasını sağlamak için kanal, WCF ' nin kanala ilişkin birden çok yönetim ortamı yaratmasını sağlar. İsteklerin işlenmesi için kullanılabilir olan kanal eşgörünümlerinin sayısı, `MaxConcurrentCalls`bağ tanımlama özelliği tarafından denetlenir. Daha fazla bilgi için bkz. “WCF bağ tanımı yapılanış seçenekleri” sayfa 584
- Güvenli teslim işlevi, hem IOperationInvoker hem de IErrorHandler WCF genişletilebilirlik noktalarını kullanır. Bu genişletilebilirlik noktaları bir uygulama tarafından dışarıdan kullanılırsa, uygulamanın önceden kayıtlı genişletilebilirlik noktalarının çağrıldığından emin olması gerekir. IErrorHandler için bu işlemi yapmamanız, bildirilmemiş hatalarla sonuçlanabilir. Failure to do so for IOperationInvoker can cause WCF to stop responding.

WCF özel kanal güvenliği

WCF için WebSphere MQ özel kanalı, kuyruk yöneticisine yalnızca yönetilmeyen istemci bağlantıları için SSL kullanımını destekler.

SSL, aşağıdaki iki yoldan biriyle belirtilebilir:

- JMS URI üzerinde SOAP üzerinde doğrudan SSL ' yi belirtin. SSL seçeneklerine ilişkin tam açıklama için bkz. SOAP için SSL ve WebSphere MQ iletimi
- İstemci kanal tanımlama çizelgesinde (CCDT) bir giriş kullanarak SSL ' yi belirtin. CCDTs ile ilgili ek bilgi için Client channel definition tablebaşlıklı konuya bakın.

WCF istemci kanal tanımlama çizelgeleri (CCDT)

WCF için WebSphere MQ özel kanalı, istemci bağlantılarına ilişkin bağlantı bilgilerini yapılandırmak için istemci kanal tanımlama çizelgelerinin (CCDT) kullanımını destekler.

CCDT ' ler bu iki ortam değişkeni aracılığıyla denetlenir:

- *MQCHLLIB* , çizelgenin bulunduğu dizini belirtir.
- *MQCHLTAB* , çizelgenin dosya adını belirtir.

Kanal tanımlama çizelgesini, SOAP üzerinden SOAP URI 'sında doğrudan belirtemezsiniz. Bu ortam değişkenleri tanımlandıysa, URI ' de belirtilen istemci bağlantısı ayrıntılarının üzerinde önceliğe sahip olur.

İstemci kanal tanımlama çizelgelerine ilişkin ek bilgi için Client channel definition table başlıklı konuya bakın.

İlgili kavramlar

İstemci kanal tanımlama çizelgesi

WCF özel kanal zehirli iletileri

Bir hizmet bir istek iletisini işlerken başarısız olduğunda ya da yanıt kuyruğuna yanıt iletisi gönderemediğinde, ileti bir zehir iletisi olarak işlenir.

Zehir isteği iletileri

Bir istek iletisi işlenemezse, bu ileti bir zehir iletisi olarak işlem görür. Bu işlem, hizmetin yeniden işlenemeyen aynı iletiyi almasını engeller. İşlenemeyen bir istek iletisinin bir zehir iletisi olarak işlenmesini sağlamak için aşağıdaki durumlardan biri doğru olmalıdır:

- Geri çıkış sayısı, istek kuyruğunda belirtilen geriletme eşiğini aştı; bu, yalnızca hizmet için güvenli teslim belirtildiğinde ortaya çıkar. Güvenli teslimata ilişkin daha fazla bilgi için bkz. "WCF için özel kanal garantili teslim" sayfa 576
- İleti doğru biçimlendirilmedi ve JMS iletisi üzerinden bir SOAP iletisi olarak yorumlanamadı.

Zehirli yanıt iletileri

Bir hizmet yanıt kuyruğuna bir yanıt iletisi gönderemezse, yanıt iletisi bir zehir iletisi olarak işlem görür. Yanıt iletileri için bu işlem, yanıt iletilerinin daha sonra yardım sorununun belirlenmesi için alınmasını sağlar.

Zehirli ileti işleme

Bir zehir iletisi için alınan işlem, kuyruk yöneticisi yapılanışına ve iletinin rapor seçeneklerinde belirlenen değerlere bağlıdır. JMS üzerinden SOAP için, istek iletilerine varsayılan olarak aşağıdaki rapor seçenekleri ayarlanır ve yapılandırılamaz:

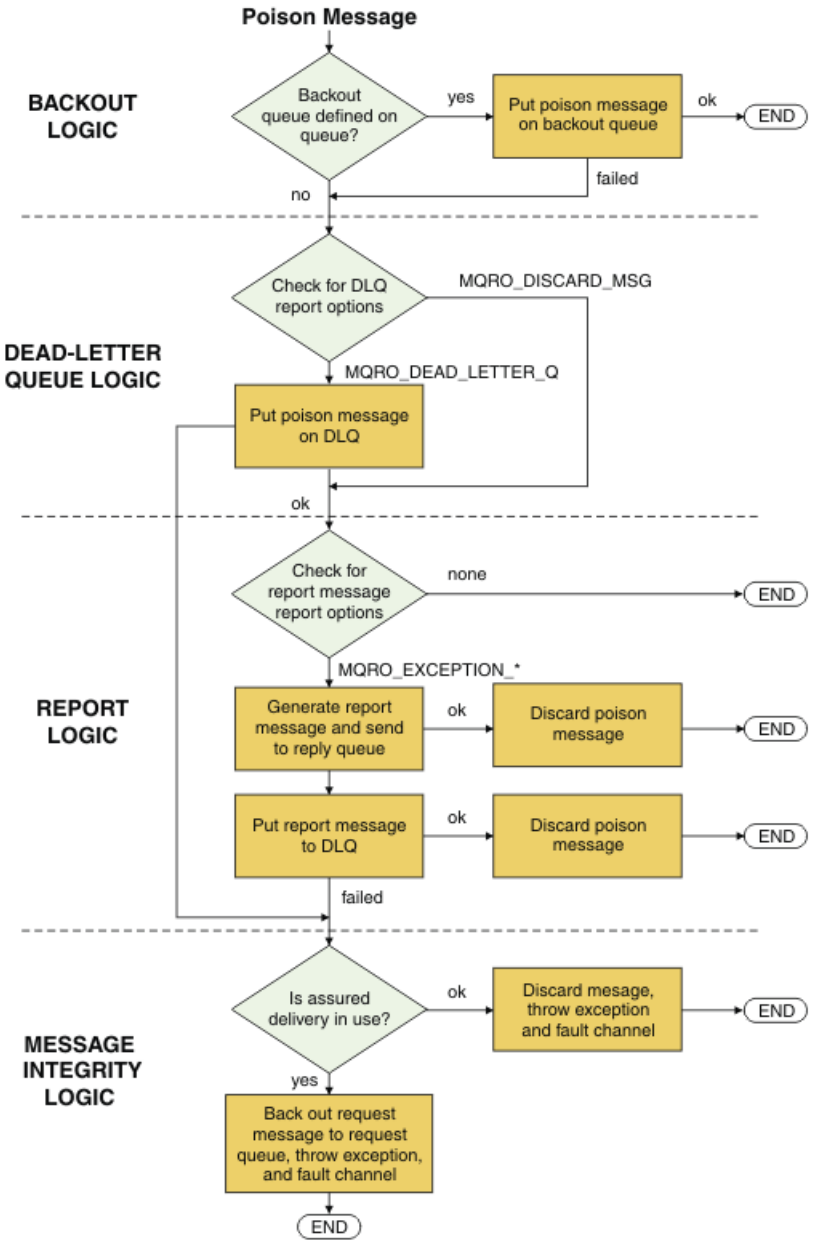
- MQRO_EXCEPTION_WITH_FULL_DATA
- MQRO_EXPIRATION_WITH_FULL_DATA
- MQRO_DISCARD_MSG

JMS üzerinden SOAP için, varsayılan olarak yanıt iletileri üzerinde aşağıdaki rapor seçeneği ayarlanır ve yapılandırılamaz:

- MQRO_DEAD_LETTER_Q

İleti, WCF olmayan bir kaynaktan geldiyse, o kaynağa ilişkin belgelere bakın.

Aşağıdaki çizge, olası eylemleri ve bir zehir ileti işleme başarısız olduğunda atılan adımları göstermektedir:



WCF bağlantısı seçenekleri

Bir WebSphere MQ özel kanalını WCF için kuyruk yöneticisine bağlamanın üç kipi vardır. Gereksinimlerinize en uygun bağlantı tipini göz önünde bulundurun.

Bağlantı seçeneklerine ilişkin ek bilgi için bkz. “Bağlantı farkları” sayfa 552

WCF mimarisi hakkında daha fazla bilgi için bkz. “WCF mimarisi” sayfa 574

Yönetilmeyen istemci bağlantısı

Bu kipte yapılan bir bağlantı, yerel makineden ya da uzak bir makinede çalışan bir WebSphere MQ Server sunucusuna WebSphere MQ istemcisi olarak bağlanır.

To use the WebSphere MQ custom channel for WCF as a WebSphere MQ client, you can install it, with the WebSphere MQ MQI client, either on the WebSphere MQ server, or on a separate machine.

Yönetilmeyen sunucu bağlantısı

Sunucu bağ tanımları kipinde kullanıldığında, WCF için WebSphere MQ özel kanalı, ağ üzerinden iletişim kurmak yerine kuyruk yöneticisi API 'sini kullanır. Bağ tanımları bağlantılarının kullanılması, WebSphere MQ uygulamaları için ağ bağlantılarını kullanmaktan daha iyi performans sağlar.

To use the bindings connection, you must install the WebSphere MQ custom channel for WCF on the WebSphere MQ server.

Yönetilen istemci bağlantısı

Bu kipte yapılan bir bağlantı, yerel makineden ya da uzak bir makinede çalışan bir WebSphere MQ Server sunucusuna WebSphere MQ istemcisi olarak bağlanır.

Bu kipe bağlanan .NET 3 için WebSphere MQ özel kanal sınıfları .NET yönetilen kodunda kalır ve yerel hizmetlere çağrılar yapmamaz. Yönetilen kod hakkında daha fazla bilgi için Microsoft belgelerine bakın.

Yönetilen istemciyi kullanmak için bir dizi sınırlama vardır. Bu sınırlamalarla ilgili daha fazla bilgi için bkz. “Yönetilen istemci bağlantıları” sayfa 552.

WCF için WebSphere MQ özel kanalının yaratılması ve yapılandırılması

WCF için WebSphere MQ V7 özel kanalları, Microsoft' un sunduğu iletim WCF kanallarıyla aynı şekilde çalışır. WCF için WebSphere MQ özel kanalı iki şekilde yaratılabilir.

Bu görev hakkında

WebSphere MQ özel kanalı WCF iletim kanalı olarak WCF ile tümleşir ve bir ileti kodlayıcı ve isteğe bağlı iletişim kuralı kanallarıyla eşlenmiş olmalıdır; böylece, uygulama tarafından kullanılabilecek tam bir kanal yığını yaratabilir. Tam kanal yığınının başarıyla yaratılmasına ilişkin iki öğe gereklidir:

1. Bağlama tanımlaması: İletim kanalı, ileti kodlayıcı ve tüm protokoller ve genel yapılandırma ayarları da içinde olmak üzere, uygulama kanalı yığınını oluşturmak için hangi öğelerin gerekli olduğunu belirtir. Özel kanal için, bağ tanımı tanımlamasının bir WCF özel bağlaması biçiminde yaratılması gerekir.
2. Uç nokta tanımlaması: Hizmet sözleşmesini bağ tanımı tanımlamasıyla bağlar ve uygulamanın bağlanabileceği yeri tanımlayan gerçek bağlantı URI 'sini de sağlar. Özel kanal için URI, JMS üzerinde SOAP URI biçiminde bir SOAP biçimidir.

Bu tanımlamalar farklı iki yoldan biriyle yaratılabilir:

- Yönetimsel olarak; tanımlar, bir uygulama yapılandırma dosyasında (örneğin: `app.config`) ayrıntılar sağlanarak oluşturulur.
- Programlı olarak; tanımlamalar doğrudan uygulama kodundan yaratılır.

Tanımları yaratmak için kullanılacak yöntemin, uygulamanın gereklerine göre aşağıdaki gibi olması gerekir:

- Yapılandırma için Yönetimle görevli yöntem, uygulamayı yeniden oluşturmadan hizmetin ayrıntılarını ve istemci konuşlandırmasını değiştirme esnekliğini sağlar.
- Yapılandırmaya ilişkin Programmatik yöntem, yapılandırma hatalarından daha fazla koruma sağlar ve yürütme sırasında dinamik olarak bir yapılandırma oluşturma yeteneği sağlar.

Bir uygulama yapılandırma dosyasında bağ tanımı ve uç nokta bilgileri sağlayarak bir WCF özel kanal yönetimi yaratılması

WCF için WebSphere MQ özel kanalı bir iletim düzeyi WCF kanalıdır. Bir uç nokta ve bağ tanımı, özel kanalı kullanmak için tanımlanmalıdır ve bu tanımlar, bir uygulama yapılandırma dosyasında bağ tanımı ve uç nokta bilgileri sağlanarak yapılabilir.

Bir iletim düzeyi WCF kanalı olan WCF için WebSphere MQ özel kanalını yapılandırmak ve kullanmak için, bir bağ tanımı ve bir uç nokta tanımlaması tanımlanmalıdır. Bağlama, kanala ilişkin yapılandırma bilgilerini bulundurur ve uç nokta tanımlaması bağlantı ayrıntılarını içerir. Bu tanımlamalar iki şekilde yaratılabilir:

- Burada açıklandığı gibi, doğrudan uygulama kodundan programlı olarak: “Bağ tanımı ve uç noktası bilgilerini programlı olarak gizleyerek bir WCF özel kanalı yaratılması” sayfa 582
- Aşağıdaki yordamda açıklandığı gibi, ayrıntıları bir uygulama yapılandırma dosyasında sağlayarak yönetimsel olarak.

İstemci ya da hizmet uygulaması yapılanış kütüğü yaygın olarak *yourappname*`.exe.config` adını taşır; burada *yakınadı* uygulamanızın adıdır. Uygulama yapılandırma dosyası, Microsoft hizmet yapılandırması düzenleyicisi aracı `SvcConfigEditor.exe` adlı aracı aşağıdaki şekilde kullanarak en kolay şekilde değiştirilir:

- `SvcConfigEditor.exe` yapılandırma düzenleyicisi aracını başlatın. Aracın varsayılan kuruluş yeri şudur: *`Drive:`*`\Program Files\Microsoft SDKs\Windows\v6.0\Bin\SvcConfigEditor.exe` (burada *Sürücü:* , kuruluş sürücüsünün adıdır).

Adım 1: WCF ' nin özel kanalı bulmasını sağlamak için bir bağlama öğesi uzantısı ekleyin

1. Menüyü açmak için **Gelişmiş** > **Uzantı** > **bağlayıcı öğesi** seçeneğini sağ tıklatın ve **Yeni**seçeneğini belirleyin.
2. Bu çizelgede gösterildiği gibi, alanları doldurun:

Çizelge 73. Yeni bağ tanımlama öğesi alanları

Alan	**Değer**
Ad	`IBM.XMS.WCF.SoapJmsIbmTransportChannel`
Tür	Genel Assembly Cache (GAC) içinde `IBM.XMS.WCF.dll` seçeneğine gidin ve `IBM.XMS.WCFSoapJmsIbmTransportBindingElementConfig`seçeneğini belirleyin.

Adım 2: Bir WCF ileti kodlayıcısıyla özel kanalı çiftleyen özel bir bağ tanımı tanımlaması yaratın.

1. Menüyü açmak için **Bağlamalar** 'ı sağ tıklatın ve **Yeni Bağ Tanımı Yapılandırması**' yı seçin.
2. Bu çizelgede gösterildiği gibi, alanları doldurun:

Çizelge 74. Yeni bağ tanımlama yapılanış alanları

Alan	**Değer**
Ad	`CustomBinding_WMQ`
BindingElement 1	`textMessageEncoding (MessageVersion: Soap11)`
BindingElement 2	`IBM.XMS.WCF.SoapJmsIbmTransportChannel`

Adım 3: Bağ tanımlama özelliklerini belirtin

1. *IBM.XMS.WCF.SoapJmsIbmTransportChannel* : “Adım 2: Bir WCF ileti kodlayıcısıyla özel kanalı çiftleyen özel bir bağ tanımı tanımlaması yaratın.” sayfa 581içinde oluşturduğunuz bağlayıcıdan bağlama bağlanması.
2. Aşağıdaki yerde açıklandığı gibi, özelliklerin varsayılan değerlerinde gerekli değişiklikleri yapın: “WCF bağ tanımı yapılanış seçenekleri” sayfa 584

Adım 4: Bir uç nokta tanımlaması yaratın

Create an endpoint definition which references the custom binding you created in: "Adım 2: Bir WCF ileti kodlayıcısıyla özel kanalı çiftleyen özel bir bağ tanımı tanımlaması yaratın." sayfa 581 and provides the connection details of the service. Bu bilgilerin belirtilme şekli, tanımın bir istemci uygulaması ya da hizmet uygulaması olup olmadığına bağlıdır.

Bir istemci uygulaması için, istemci kısmına aşağıdaki gibi bir uç nokta tanımlaması ekleyin:

1. Menüyü açmak için **İstemci** > **Uç Noktalar** seçeneğini sağ tıklatın ve **Yeni İstemci Uç Noktası**' yı seçin.
2. Bu çizelgede gösterildiği gibi, alanları doldurun:

Çizelge 75. Yeni istemci uç noktası alanları

Alan	Değer
Ad	`Endpoint_WMQ`
Adres	*Hizmet için erişmek için gerekli WMQ bağlantı ayrıntılarını açıklayan SOAP/JMS URI.* Daha fazla ayrıntı için bkz. "WebSphere MQ WCF uç noktası URI adresi biçimi için özel kanal" sayfa 583
Bağ Tanımı	`customBinding`
BindingConfiguration	`CustomBinding_WMQ`
Sözleşme	*Hizmet sözleşmesi arabiriminizin adı*

Bir hizmet uygulaması için, hizmetler bölümüne aşağıdaki gibi bir hizmet tanımı ekleyin:

1. Menüyü açmak için **Hizmetler** seçeneğini sağ tıklatın ve **Yeni Hizmet**seçeneğini belirleyin ve daha sonra, barındırılacak hizmet sınıfını seçin.
2. Yeni hizmetinize ilişkin **Uç Noktalar** bölümüne bir uç nokta tanımlaması ekleyin ve bu çizelgede gösterildiği gibi, alanları doldurun:

Çizelge 76. Yeni hizmet uç noktası alanları

Alan	Değer
Ad	`Endpoint_WMQ`
Adres	*Hizmet için erişmek için gerekli WMQ bağlantı ayrıntılarını açıklayan SOAP/JMS URI.* Daha fazla ayrıntı için bkz. "WebSphere MQ WCF uç noktası URI adresi biçimi için özel kanal" sayfa 583
Bağ Tanımı	`customBinding`
BindingConfiguration	`CustomBinding_WMQ`
Sözleşme	*Hizmet gerçekleştirme sınıfınızın adı*

Bağ tanımı ve uç noktası bilgilerini programlı olarak gizleyerek bir WCF özel kanalı yaratılması

WCF için WebSphere MQ özel kanalı bir iletim düzeyi WCF kanalıdır. Bir uç nokta ve bağ tanımı, özel kanalı kullanmak için tanımlanmalıdır ve bu tanımlar doğrudan uygulama kodundan programlanabilir.

Bir iletim düzeyi WCF kanalı olan WCF için WebSphere MQ özel kanalını yapılandırmak ve kullanmak için, bir bağ tanımı ve bir uç nokta tanımlaması tanımlanmalıdır. Bağlama, kanala ilişkin yapılandırma bilgilerini bulundurur ve uç nokta tanımlaması bağlantı ayrıntılarını içerir. Daha fazla bilgi için bkz. "WCF örneklerinin kullanılması" sayfa 591

Bu tanımlamalar iki şekilde yaratılabilir:

- Burada açıklandığı gibi, bir uygulama yapılandırma dosyasında ayrıntıları sağlayarak yönetimsel olarak: “Bir uygulama yapılandırma dosyasında bağ tanımı ve uç nokta bilgileri sağlayarak bir WCF özel kanal yönetimi yaratılması” sayfa 580
- Aşağıdaki örnekte açıklandığı gibi, doğrudan uygulama kodundan programsal olarak.

Adım 1: Kanalın iletim bağ tanımı öğesinin bir eşgörünümünü yaratın.

Uygulamanınıza aşağıdaki kodu ekleyin:

```
SoapJmsIbmTransportBindingElement transportBindingElement = new
SoapJmsIbmTransportBindingElement();
```

Adım 2: Bağ tanımlama özelliklerinin ayarlanması

Set any required binding properties, for example, by adding the following code to your application to set the `ClientConnectionMode`.

```
transportBindingElement.ClientConnectionMode = XmsWCFBindingProperty.AS_URI;
```

Adım 3: İletim kanalını bir ileti kodlayıcısıyla çiftleyen özel bir bağ tanımı yaratın.

Uygulamanınıza aşağıdaki kodu ekleyerek özel bir bağ tanımı yaratın:

```
Binding binding = new CustomBinding(new TextMessageEncodingBindingElement(),
transportBindingElement);
```

Adım 4: SOAP/JMS URI 'si yaratılıyor

Hizmete erişmek için gereken WebSphere MQ bağlantı ayrıntılarını tanımlayan SOAP/JMS URI 'si, uç nokta adresi olarak sağlanmalıdır. Bu, kanalın bir hizmet uygulaması ya da istemci uygulaması için kullanılıp kullanılmadığına bağlıdır.

İstemci uygulamaları için, SOAP/JMS URI 'si aşağıdaki gibi bir `EndpointAddress` olarak yaratılmalıdır:

```
EndpointAddress address = new EndpointAddress("jms:/queue?
destination=SampleQ@QM1&connectionFactory=connectQueueManager(QM1)&initialContextFactory=com.ibm
.mq.jms.Nojndi");
```

Hizmet uygulamaları için, SOAP/JMS URI 'si aşağıdaki gibi bir URI olarak yaratılmalıdır:

```
Uri address = new Uri("jms:/queue?
destination=SampleQ@QM1&connectionFactory=connectQueueManager(QM1)&initialContextFactory=com.ibm
.mq.jms.Nojndi");
```

Uç nokta adresiyle ilgili daha fazla bilgi için bkz. “WebSphere MQ WCF uç noktası URI adresi biçimi için özel kanal” sayfa 583

WebSphere MQ WCF uç noktası URI adresi biçimi için özel kanal

Bir Evrensel Kaynak Tanıtıcısı (URI), bir web hizmeti belirtmek için konum ve bağlantı ayrıntıları sağlar. Bu URI biçimi, hedef hizmetlere erişirken SOAP/ WebSphere MQ' ya özgü parametreler ve seçenekler üzerinden kapsamlı bir denetim derecelerine izin verir.

Bir web hizmeti URI (Universal Resource Identifier; Evrensel Kaynak Tanıtıcısı) kullanılarak belirtilir. Bu bölüm, SOAP için WebSphere MQ iletimi içinde desteklenen URI biçimini belirtir. Bu URI biçimi, hedef hizmetlere erişirken SOAP/WebSphere MQ' ya özgü parametreler ve seçenekler üzerinden kapsamlı bir denetim derecelerine izin verir. Bu biçim, WebSphere Application Server (WAS) ile ve CICS ile WebSphere MQ olanağının her iki ürünle tümleşmesini kolaylaştırmaktadır.

URI sözdizimi aşağıdaki gibidir:

```
jms:/queue?name=value&name=value...
```

Burada `name` , bir parametre adı ve *value* uygun bir değerdir ve `name=`*value* öğesi, ikinci ve sonraki oluşumlardan önce ve işareti (&) ile herhangi bir sayıda yinelenebilir.

URI özelliklerini ayarlama hakkında daha fazla bilgi için bkz: Web hizmeti konuşlandırması için URI sözdizimi ve parametreleri

Parametre adları büyük ve küçük harfe duyarlıdır, WebSphere MQ nesnelerinin adları gibi. Herhangi bir parametre bir kereden fazla belirtilirse, parametrenin son oluşumu, istemci uygulamalarının URI ' ye sonuna ekleyerek parametre değerlerini geçersiz kılabileceği anlamına gelir. Ek olarak tanınmayan herhangi bir parametre varsa, bunlar yoksayılır.

Bir URI ' yi bir XML dizgisinde saklıyorsa, ve imi karakterini "&" olarak göstermeniz gerekir. Benzer şekilde, bir URI bir komut dosyasında kodlandıysa, tersi durumda kabuk tarafından yorumlanacak **&** gibi çıkış karakterlerine özen gösteriniz.

Bu, bir Axis hizmeti için basit bir URI örneğidir:

```
jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi
```

.NET hizmeti için basit bir URI örneğidir:

```
jms:/queue?destination=myQ&connectionFactory=()&targetService=MyService.asmx
&initialContextFactory=com.ibm.mq.jms.Nojndi
```

Yalnızca gerekli parametreler sağlanır (yalnızca .NET hizmetleri için`targetService` gereklidir) ve `connectionFactory` ' a seçenek verilmez.

Bu Eksen örneğinde, `connectionFactory` bir dizi seçenek içerir:

```
jms:/queue?destination=myQ@myRQM&connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
&initialContextFactory=com.ibm.mq.jms.Nojndi
```

Bu Eksen örneğinde, `connectionFactory` ürününün `sslPeerName` seçeneği de belirtilmiş. sslPeerName 'in değeri, ad değer çiftlerini ve önemli gömülü boşlukları içerir:

```
jms:/queue?destination=myQ@myRQM&connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
sslPeerName(CN=MQ Test 1,O=IBM,S=Hampshire,C=GB)
&initialContextFactory=com.ibm.mq.jms.Nojndi
```

WCF bağ tanımı yapılanış seçenekleri

Bu konuda, yapılandırma seçeneklerinin özel kanallar bağlama bilgilerine nasıl uygulanabileceği ve kullanılabilir seçeneklerin listelendiği açıklanmaktadır.

Bağ tanımlama yapılanışı seçenekleri iki farklı yoldan biriyle ayarlanabilir:

1. Yönetimsel olarak: Bağlama özelliği ayarlarının, uygulamalar yapılandırma dosyasındaki özel bağlama tanımlamasının iletim kısmında belirtilmesi gerekir; örneğin: `app.config`
2. Programlı olarak: Özel bağ tanımının kullanıma hazırlanması sırasında özelliği belirtmek için uygulama kodunun değiştirilmesi gerekir.

Bağ tanımlama özelliklerinin yönetimsel olarak ayarlanması

Bağlama özelliği ayarları, uygulama yapılandırma dosyasında da belirtilebilir; örneğin: `app.config`. Yapılandırma dosyası **`svcutil`**tarafından oluşturulur, örneğin:

```
<customBinding>

...

     <IBM.XMS.WCF.SoapJmsIbmTransportChannel maxBufferPoolSize="524288"
      maxMessageSize="4000000" clientConnectionMode="0" maxConcurrentCalls="16"/>
```

```
...
</customBinding>
```

Bağ tanımlama özelliklerinin programsal olarak ayarlanması

İstemci bağlantı kipini belirtmek üzere bir WCF bağ tanımı özelliği eklemek için, özel bağ tanımının kullanıma hazırlanması sırasında özelliği belirtmek için hizmet kodunu değiştirmeniz gerekir.

Yönetilmeyen istemci bağlantı kipini belirtmek için aşağıdaki örneği kullanın:

```
SoapJmsIbmTransportBindingElement transportBindingElement = new
SoapJmsIbmTransportBindingElement();
transportBindingElement.ClientConnectionMode = XmsWCFBindingProperty.CLIENT_UNMANAGED;

Binding sampleBinding = new CustomBinding(new TextMessageEncodingBindingElement(),
                                                    transportBindingElement);
```

WCF bağ tanımı özellikleri

Özellik adı	İstemci ya da Hizmet uygulaması	Yönetim değeri	Programlı değer	Tanım
maxBufferPoolSize	Her ikisi	0-64 bit işaretli tamsayı	0-64 bit işaretli tamsayı	Kanal örneğine ilişkin WCF ileti arabelleklerini saklamak için kullanılabilecek bellek büyüklüğü üst sınırını belirler.
maxMessageBoyutu	Her ikisi	1-32 bit işaretli tamsayı	1-32 bit işaretli tamsayı	Tek bir WCF iletisi için kullanılabilecek bellek üst sınırını belirler.
clientConnectionKipi	Her ikisi	0 (Varsayılan değer) 1	AS_URI (Varsayılan değer) CLIENT_UNMANAGED	İletim kanalının istemci bağlantı kipini belirtir. 0 , istemci bağlantı kipinin URI 'de belirtildiği gibi olduğu anlamına gelir. Yalnızca istemci bağlantısı kullanılırsa kullanılır. İstemci bağlantı kipinin URI 'de belirtildiği gibi olduğunu belirtir. İstemci bağlantı kipi belirlenmezse, varsayılan değer 0 olur. 1 , istemci bağlantı kipinin yönetilmeyen bir istemci olduğu anlamına gelir. Yalnızca istemci bağlantısı kullanılırsa kullanılır.

Özellik adı	İstemci ya da Hizmet uygulaması	Yönetim değeri	Programlı değer	Tanım
MaxConcurrentÇağrıları	İstemci	menzil 0-2 147 483 647 Varsayılan değer 16 'tır	menzil 0-2 147 483 647 Varsayılan değer 16 'tır	Bu özellik, herhangi bir zamanda tek bir istemci yetkili sunucusunda yer alabilen koşutzamanlı işlem sayısı üst sınırını tanımlar. Daha fazla işlem başlatılırsa, bunlar, devam eden bir işlem tamamlanıncaya kadar ya da zamanaşımına kadar kuyruğa alınır. Bu ayar, tek bir yetkili sunucu tarafından tüketilebilecek iş parçacıklarını ve kaynakları denetlemek için kullanılabilir. `0` , bu sınırı kaldırır ve tüm işlemlerin eşzamanlı olarak denenmesini sağlar.
MaxConcurrentÇağrıları	Hizmet	menzil 1-2 147 483 647 Varsayılan değer 16 'tır	menzil 1-2 147 483 647 Varsayılan değer 16 'tır	Bu özellik, yalnızca güvenli teslim özelliği etkinleştirilmişse kullanılır (güvenli teslimata ilişkin daha fazla bilgi için bkz. “WCF için özel kanal garantili teslim” sayfa 576). Verili uç nokta için aynı anda devam edebilen koşutzamanlı işlem sayısı üst sınırını belirtir. Bu ayarı değiştirirken dikkatli olun. Eşzamanlı işlemlerin her biri, özel kanalın yeni bir eşgörünümü ve iş parçacığı havuzundaki ilişkili iş parçacıklarının istekleri işleme almak için ek kaynakları gerektirir. Aşırı ayırma işlemi, üretkenliğinizi etkileyebilir ve performansı ağır bir şekilde etkileyebilir. Bu özelliği desteklemek için iş parçacığı havuzunun uygun yapılandırması yapılmalıdır.

WCF için hizmetler oluşturuluyor ve barındırılıyor

WCF hizmetlerini nasıl yaratacağını ve yapılandıracağını açıklayan Microsoft Windows Communication Foundation (WCF) hizmetlerine genel bakış.

WCF için IBM WebSphere MQ özel kanalı ve bunu kullanan WCF hizmetleri aşağıdaki yöntemlerle barındırılabilir:

- Kendi kendine barındırma
- Windows Hizmeti

WCF için IBM WebSphere MQ özel kanalı Windows Process Activation Service içinde barındırılamaz.

Aşağıdaki konularda, ilgili adımları göstermek için bazı basit kendi kendine barındırma örnekleri sağlanmaktadır. Daha fazla bilgi ve en son ayrıntılar içeren Microsoft WCF çevrimiçi belgeleri, https://msdn.microsoft.comadresindeki Microsoft MSDN web sitesinde bulunabilir.

1. yöntemi kullanarak WCF hizmeti uygulamaları oluşturuluyor: Bir uygulama yapılandırma dosyası kullanılarak otomatik olarak barındırma

Bir uygulama yapılandırma dosyası oluşturduktan sonra, hizmetin bir eşgörünümünü açın ve belirtilen kodu uygulamanızın üzerine ekleyin.

Başlamadan önce

Create or edit an application configuration file for the service, as described in: "Bir uygulama yapılandırma dosyasında bağ tanımı ve uç nokta bilgileri sağlayarak bir WCF özel kanal yönetimi yaratılması" sayfa 580

Bu görev hakkında

1. Hizmet anasisteminde hizmetin bir eşgörünümünü somutlaştırır ve açar. Hizmet tipi, hizmet yapılanış kütüğünde belirtilen hizmet tipiyle aynı olmalıdır.
2. Uygulamanınıza aşağıdaki kodu ekleyin:

```
ServiceHost service = new ServiceHost(typeof(MyService));
service.Open();
...
service.Close();
```

2. yöntemi kullanarak WCF hizmeti uygulamaları oluşturuluyor: Kendi kendine barındırma (self-hosting) programlı olarak doğrudan uygulamadan

Bağ tanımlama özelliklerini ekleyin, hizmet anasistemini gerekli hizmet sınıfının bir somut örneğiyle yaratın ve hizmeti açın.

Başlamadan önce

1. Proje için özel kanal `IBM.XMS.WCF.dll` dosyasına bir başvuru ekleyin. `IBM.XMS.WCF.dll` , *`WMQInstallDir`*`\bin` içinde *WMQInstallDir* , WebSphere MQ 7 'nin kurulu olduğu dizindir.
2. `IBM.XMS.WCF` ad alanına bir *using* deyimi ekleyin, örneğin: `using IBM.XMS.WCF`
3. Aşağıdaki yerde açıklandığı gibi, kanal bağ tanımı öğesinin ve uç noktasının bir eşgörünümünü oluşturun: "Bağ tanımı ve uç noktası bilgilerini programlı olarak gizleyerek bir WCF özel kanalı yaratılması" sayfa 582

Bu görev hakkında

Kanalın bağ tanımı özelliklerinde değişiklik yapılması gerekiyorsa, aşağıdaki adımları tamamlayın:

1. Bağlama özelliklerini aşağıdaki örnekte gösterildiği gibi `transportBindingElement` olarak ekleyin:

```
SoapJmsIbmTransportBindingElement transportBindingElement = new
SoapJmsIbmTransportBindingElement();
Binding binding = new CustomBinding(new TextMessageEncodingBindingElement(),
transportBindingElement);
Uri address = new Uri("jms:/queue?destination=SampleQ@QM1&connectionFactory=

connectQueueManager(QM1)&initialContextFactory=com.ibm.mq.jms.Nojndi");
```

2. Hizmet anasistemini, gerekli hizmet sınıfının bir somut örneğiyle yaratın:

```
ServiceHost service = new ServiceHost(typeof(MyService));
```

3. Hizmeti açın:

```
service.AddServiceEndpoint(typeof(IMyServiceContract), binding, address);
service.Open();
```

```
    ...
service.Close();
```

HTTP uç noktası kullanılarak meta verilerin gösterilmesi

WCF için WebSphere MQ özel kanalını kullanmak üzere yapılandırılmış bir hizmetin meta verilerinin gösterilmesine ilişkin yönergeler.

Bu görev hakkında

Hizmetler meta verileri açıksa (örneğin, `svcutil` gibi araçların, örneğin, çevrimdışı WSDL dosyasından değil, doğrudan çalışan hizmetten erişebilmesi için), HTTP uç noktası ile hizmet meta verilerinin gösterilmesiyle gerçekleştirilmelidir. Bu ek uç noktayı eklemek için aşağıdaki adımlar kullanılabilir.

1. Meta verilerin ServiceHost' e gösterilmesi gereken temel adresi ekleyin, örneğin:

```
ServiceHost service = new ServiceHost(typeof(TestService),
                      new Uri("http://localhost:8000/MyService"));
```

2. Hizmet açılmadan önce aşağıdaki kodu ServiceHost ' a ekleyin:

```
ServiceMetadataBehavior metadataBehavior = new ServiceMetadataBehavior();
  metadataBehavior.HttpGetEnabled = true;
  service.Description.Behaviors.Add(metadataBehavior);
  service.AddServiceEndpoint(typeof(IMetadataExchange),
     MetadataExchangeBindings.CreateMexHttpBinding(), "mex");
```

Sonuçlar

Meta veriler şu anda şu adreste bulunur: `http://localhost:8000/MyService`

WCF için istemci uygulamaları oluşturuluyor

Microsoft Windows Communication Foundation (WCF) istemci uygulamalarının oluşturulmasına ve oluşturulmasına genel bakış.

Bir WCF hizmeti için bir istemci uygulaması yaratılabilir; istemci uygulamaları genellikle uygulama tarafından kullanılabilecek gereken yapılandırma ve yetkili sunucu dosyalarını yaratmak için Microsoft ServiceModel Metadata Utility Tool (Svcutil.exe) kullanılarak oluşturulur.

Çalışan bir hizmetteki meta verileri içeren svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılandırma dosyaları oluşturulması

WCF için WebSphere MQ özel kanalını kullanmak üzere yapılandırılmış bir hizmet için istemci oluşturmak üzere Microsoft svcutil.exe aracını kullanma yönergeleri.

Başlamadan önce

Doğrudan uygulama tarafından kullanılabilecek zorunlu yapılandırma ve yetkili sunucu dosyalarına yaratmak için svcutil aracını kullanmak için üç önkoşul vardır:

- svcutil aracı başlatılmadan önce WCF hizmeti çalışıyor olmalıdır.
- WCF hizmeti, doğrudan çalışan bir hizmetten istemci oluşturmak için WebSphere MQ özel kanal uç noktası başvurularına ek olarak bir HTTP kapısı kullanarak meta verilerini açıklamalıdır.
- svcutil için yapılandırma verilerinde özel kanal kaydedilmelidir.

Bu görev hakkında

Aşağıdaki adımlar, WebSphere MQ özel kanalını kullanmak üzere yapılandırılmış bir hizmet için istemci oluşturulmasını açıklar, ancak aynı zamanda ayrı bir HTTP kapısı aracılığıyla çalışma zamanında meta verilerini de gösterir:

1. WCF hizmetini başlatın (hizmet, svcutil aracı başlatılmadan önce çalışır durumda olmalıdır).
2. Add the details from the `svcutil.exe` config file from the root of the installation, into the active svcutil configuration file, typically `C:\Program Files\Microsoft`

SDKs\Windows\v6.0A\bin\svcutil.exe.config so svcutil recognizes the WebSphere MQ custom channel.

3. Bir komut isteminden svcutil komutunu çalıştırın; örneğin:

```
svcutil /language:C# /r:<installlocation>\bin\IBM.XMS.WCF.dll
               /config:app.config http://localhost:8000/IBM.XMS.WCF/samples
```

4. Oluşturulan app.config ve YourService.cs dosyalarını, Microsoft Visual Studio istemcisi projesine kopyalayın.

Sonraki adım

Hizmet meta verileri doğrudan alınamazsa, istemci dosyalarını wsdl yerine wsdl 'den oluşturmak için svcutil kullanılabilir. Daha fazla bilgi için bkz. “WSDL ile svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılanışı dosyaları oluşturuluyor” sayfa 589

WSDL ile svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılanışı dosyaları oluşturuluyor

Hizmetin meta verileri kullanılamazsa, WSDL ' den WCF istemcileri oluşturmaya ilişkin yönergeler.

Bir çalışan hizmetten meta verilerden istemci oluşturmak için hizmetin meta verileri doğrudan alınamazsa, istemci dosyalarını WSDL ' den oluşturmak için svcutil kullanılabilir. WSDL ' nin WebSphere MQ özel kanalının kullanılacağını belirtmek için aşağıdaki değişiklikler yapılması gerekir:

1. Aşağıdaki ad alanı tanımlarını ve ilke bilgilerini ekleyin:

```
<wsdl:definitions
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">

        <wsp:Policy wsu:Id="CustomBinding_IWMQSampleContract_policy">
            <wsp:ExactlyOne>
                <wsp:All>
                    <xms:xms xmlns:xms="http://sample.schemas.ibm.com/policy/xms" />
                </wsp:All>
            </wsp:ExactlyOne>
        </wsp:Policy>

...

</wsdl:definitions>
```

2. Bağ tanımları bölümünü değiştirerek, yeni ilke kısmına bakın ve temeldeki bağ tanımlama öğesinden transport tanımını kaldırın:

```
<wsdl:definitions ...>

        <wsdl:binding ...>
                <wsp:PolicyReference URI="#CustomerBinding_IWMQSampleContract_policy" />
                <[soap]:binding ... transport="" />
                ...
        </wsdl:binding>
</wsdl:definitions>
```

3. Bir komut isteminden svcutil komutunu çalıştırın; örneğin:

```
svcutil /language:C# /r:MQ_INSTALLATION_PATH\bin\IBM.XMS.WCF.dll
        /config:app.config
MQ_INSTALLATION_PATH\src\samples\WMQAxis\default\service\soap.server.stockQuoteAxis_Wmq.wsdl
```

Burada *MQ_INSTALLATION_PATH* , WebSphere MQ' nun kuruluş dizinidir.

Uygulama yapılanış kütüğüyle istemci yetkili sunucusu kullanılarak WCF istemcisi uygulamaları oluşturulması

Başlamadan önce

Create or edit an application configuration file for the client, as described in: “Bir uygulama yapılandırma dosyasında bağ tanımı ve uç nokta bilgileri sağlayarak bir WCF özel kanal yönetimi yaratılması” sayfa 580

Bu görev hakkında

İstemci yetkili sunucusunun bir eşgörünümünü somutlaştırır ve açar. Oluşturulan yetkili sunucuya geçirilen değiştirge, istemci yapılanış kütüğünde belirtilen uç nokta adıyla (örneğin `Endpoint_WMQ`) aynı olmalıdır.

```
MyClientProxy myClient = new MyClientProxy("Endpoint_WMQ");
            try {
                   myClient.myMethod("HelloWorld!");
                   myClient.Close();
            }
            catch (TimeoutException e) {
                Console.Out.WriteLine(e);
                myClient.Abort();
            }
            catch (CommunicationException e) {
                Console.Out.WriteLine(e);
                myClient.Abort();
            }
            catch (Exception e) {
                Console.Out.WriteLine(e);
                myClient.Abort();
            }
```

Bir istemci yetkili sunucusunu programlı yapılandırmayla oluşturmak için WCF istemcisi uygulamaları oluşturulması

Başlamadan önce

1. Proje için özel kanal `IBM.XMS.WCF.dll` dosyasına bir başvuru ekleyin. `IBM.XMS.WCF.dll` , *`WMQInstallDir`*`\bin` dizininde; burada *WMQInstallDir* , WebSphere MQ 7 'nin kurulu olduğu dizindir.
2. `IBM.XMS.WCF` ad alanına bir *using* deyimi ekleyin, örneğin: `using IBM.XMS.WCF`
3. Şu yerde açıklandığı gibi, kanalın uç noktası ve uç noktasının bir eşgörünümünü oluşturun: “Bağ tanımı ve uç noktası bilgilerini programlı olarak gizleyerek bir WCF özel kanalı yaratılması” sayfa 582

Bu görev hakkında

Kanalın bağ tanımı özelliklerinde değişiklik yapılması gerekiyorsa, aşağıdaki adımları tamamlayın:

1. Bağlama özelliklerini aşağıdaki şekilde gösterildiği gibi `transportBindingElement` içine ekleyin:

   ```
   SoapJmsIbmTransportBindingElement transportBindingElement = new
   SoapJmsIbmTransportBindingElement();
   Binding binding = new CustomBinding(new TextMessageEncodingBindingElement(),
   transportBindingElement);
   EndpointAddress address =
     new EndpointAddress("jms:/queue?destination=SampleQ@QM1&connectionFactory=

   connectQueueManager(QM1)&initialContextFactory=com.ibm.mq.jms.Nojndi");
   ```

2. İstemci yetkili sunucusunu aşağıdaki şekilde gösterildiği gibi yaratın; burada *bağ tanımı* ve *uç noktası adresi* , “1” sayfa 590 adımında yapılandırılan ve aşağıdaki şekilde geçirilen uç noktası ve uç noktası adresleridir:

   ```
    MyClientProxy myClient = new MyClientProxy(binding, endpoint address);
               try {
                      myClient.myMethod("HelloWorld!");
                      myClient.Close();
               }
               catch (TimeoutException e) {
                   Console.Out.WriteLine(e);
                   myClient.Abort();
               }
               catch (CommunicationException e) {
                   Console.Out.WriteLine(e);
   ```

```
                            myClient.Abort();
        }
        catch (Exception e) {
            Console.Out.WriteLine(e);
            myClient.Abort();
        }
```

WCF örneklerinin kullanılması

Windows Communication Foundation (WCF) örnekleri, WebSphere MQ özel kanalının nasıl kullanılabileceğiyle ilgili bazı basit örnekler sağlar.

Örnek projeleri oluşturmak için, Microsoft .NET 3.5 SDK ya da Microsoft Visual Studio 2008 gereklidir.

Basit tek yönlü istemci ve sunucu WCF örneği

Bu örnek, tek yönlü kanal şeklini kullanarak bir WCF istemcisinden bir Windows Communication Foundation (WCF) hizmetini başlatmak için kullanılan WebSphere MQ özel kanalısını gösterir.

Bu görev hakkında

Hizmet, konsola bir dizgi çıktıran tek bir yöntemi uygular. The client has been generated by using the `svcutil` tool to retrieve the service metadata from a separately exposed HTTP endpoint as described in "Çalışan bir hizmetteki meta verileri içeren svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılandırma dosyaları oluşturulması" sayfa 588

Örnek, aşağıdaki yordamda açıklandığı gibi belirli kaynak adları ile yapılandırılmıştır. Kaynak adlarını değiştirmeniz gerekirse, *MQ_INSTALLATION_PATH*`\tools\dotnet\samples\cs\wcf\samples\WCF\oneway\client\app.config` dosyasında istemci uygulamasındaki karşılık gelen değeri ve *MQ_INSTALLATION_PATH*`\tools\dotnet\samples\cs\wcf\samples\WCF\oneway\service\TestServices.cs` dosyasındaki hizmet uygulamasında da değişiklik yapmak gerekir; burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQiçin kuruluş dizinidir. JMS uç noktası URI 'sını biçimlendirme hakkında daha fazla bilgi için, WebSphere MQ ürün belgelerindeki *WebSphere MQ Transport for SOAP* başlıklı konuya bakın. Örnek çözümü ve kaynağı değiştirmeniz gerekirse, örneğin, Microsoft Visual Studio 8 ya da sonraki bir IDE ' ye (örneğin,.Visual Studio 8 ya da üstü) gerek duyarsınız.

Yordam

1. *QM1*adlı bir kuyruk yöneticisi yaratın.
2. *SampleQ*adlı bir kuyruk hedefi yaratın.
3. İletişimci iletilerin beklediği şekilde hizmeti başlatın: *MQ_INSTALLATION_PATH*`\tools\dotnet\samples\cs\wcf\samples\WCF\oneway\service\bin\Release\TestService.exe` dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQiçin kuruluş dizinidir.
4. İstemciyi bir kez çalıştırın: *MQ_INSTALLATION_PATH*`\tools\dotnet\samples\cs\wcf\samples\WCF\oneway\client\bin\Release\TestClient.exe` dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQiçin kuruluş dizinidir.

   İstemci uygulaması döngüleri beş kez beş ileti göndererek *SampleQ*' a gönderilir.

Sonuçlar

The service application gets the messages from *SampleQ* and displays `Hello World` on the screen five times.

Sonraki adım

Basit istek yanıtlama istemcisi ve sunucu WCF örneği

This sample demonstrates the WebSphere MQ custom channel being used to start a Pencereler Communication foundation (WCF) service from a WCF client using a request-reply channel shape.

Bu görev hakkında

Bu hizmet, iki sayı eklemek ve çıkarması için bazı basit hesap makinesi yöntemleri sağlar ve sonucu döndürür. The client has been generated by using the `svcutil` tool to retrieve the service metadata from a separately exposed HTTP endpoint as described in "Çalışan bir hizmetteki meta verileri içeren svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılandırma dosyaları oluşturulması" sayfa 588

Örnek, açıklanan aşağıdaki yordamda olduğu gibi belirli kaynak adlarıyla yapılandırılmıştır. If you need to change the resource names, then you also need to change the corresponding value on the client application in the *MQ_INSTALLATION_PATH*`\Tools\wcf\samples\WCF\requestreply\client\app.config` file, and on the service application in the *MQ_INSTALLATION_PATH*`\Tools\wcf\samples\WCF\requestreply\service\RequestReplyService.cs` file, where *MQ_INSTALLATION_PATH* is the installation directory for WebSphere MQ. JMS uç noktası URI 'sını biçimlendirme hakkında daha fazla bilgi için, WebSphere MQ ürün belgelerindeki *WebSphere MQ Transport for SOAP* başlıklı konuya bakın. Örnek çözümü ve kaynağı değiştirmeniz gerekirse, örneğin, Microsoft Visual Studio 8 ya da sonraki bir IDE ' ye (örneğin,.Visual Studio 8 ya da üstü) gerek duyarsınız.

Yordam

1. *QM1*adlı bir kuyruk yöneticisi yaratın.
2. *SampleQ*adlı bir kuyruk hedefi yaratın.
3. *SampleReplyQ*adlı bir kuyruk hedefi yaratın.
4. İletişiminin iletileri beklediği şekilde hizmeti başlatın: *MQ_INSTALLATION_PATH*`\Tools\wcf\samples\WCF\requestreply\service\bin\Release\SimpleRequestReply_Service.exe` dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' nın kuruluş dizinidir.
5. İstemciyi bir kez çalıştırın: *MQ_INSTALLATION_PATH*`\Tools\wcf\samples\WCF\requestreply\client\bin\Release\SimpleRequestReply_Client.exe` dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' un kuruluş dizinidir.

Sonuçlar

İstemci çalıştırıldığı zaman, aşağıdaki işlem başlatılır ve dört kez yinelenir; bu nedenle, her bir yolla toplam beş ileti gönderilir:

1. İstemci, *SampleQ* ' a bir istek iletisi koyar ve yanıt bekler.
2. Hizmet, istek iletisini *SampleQ*' den alır.
3. Hizmet, iletinin içeriğini kullanarak bazı değerleri ekler ve çıkarır.
4. Daha sonra hizmet, sonuçları *SampleReplyQ*' da bir iletiye koyar ve istemcinin yeni bir ileti koymasını bekler.
5. İstemci *SampleReplyQ* ' dan iletiyi alır ve sonuçları ekranda görüntüler.

Sonraki adım

WCF istemcisi, WebSphere MQ örneği tarafından barındırılan bir .NET hizmetine

Hem .NET hem de Java için örnek istemci uygulamaları ve örnek hizmet yetkili sunucusu uygulamaları sağlanır. Örnekler, hisse senedi fiyat teklifi isteği alan bir hisse senedi senedi hizmetine dayalıdır ve hisse senedi alıntısını sağlar.

Başlamadan önce

The sample requires that the .NET SOAP over JMS service hosting environment is correctly installed and configured in WebSphere MQ and is accessible from a local queue manager. Ortamın kurulmasına ve yapılandırılmasına ilişkin bilgi için bkz. “SOAP için WebSphere MQ Web iletimi kurulması” sayfa 917

When the .NET SOAP over JMS service hosting environment is correctly installed and configured in WebSphere MQ and is accessible from a local queue manager, additional configuration steps must be completed.

1. WMQSOAP_HOME ortam değişkenini WebSphere MQ kuruluş dizinine (örneğin: `C:\Program Files\IBM\WebSphere MQ`) ayarlayın.
2. Ensure that the Java compiler `javac` is available and on the PATH.
3. `axis.jar` dosyasını, WebSphere kuruluş CD ' nin `prereqs/axis` dizininden WebSphere MQ üretim dizinine kopyalayın, örneğin: `C:\Program Files\IBM\WebSphere MQ\java\lib\soap`
4. PATH değişkenine ekleyin: *MQ_INSTALLATION_PATH*`\Java\lib` burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini temsil eder; örneğin: `C:\Program Files\IBM\WebSphere MQ`
5. Ensure that the location of .NET is specified correctly in *MQ_INSTALLATION_PATH*`\bin\amqwcallWSDL.cmd` where *MQ_INSTALLATION_PATH* represents the directory where WebSphere MQ is installed, for example: `C:\Program Files\IBM\WebSphere MQ`. .NET ' in yeri örnek olarak belirtilebilir: `set msfwdir=%ProgramFiles%\Microsoft Visual Studio .NET 2003\SDK\v1.1\Bin`

Önceki adımlar tamamlanınca, hizmeti test edin ve çalıştırın:

1. JMS üzerinde çalışan JMS çalışma dizini için SOAP ' ınıza gidin.
2. Doğrulama sınamasını çalıştırmak ve hizmet dinleyicisini çalışır durumda bırakmak için aşağıdaki komutlardan birini girin:
   - .NET için: *MQ_INSTALLATION_PATH*`\Tools\soap\samples\runivt dotnet hold` ; burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini temsil eder.
   - AXIS:*MQ_INSTALLATION_PATH*`\Tools\soap\samples\runivt Dotnet2AxisClient hold` için, burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini gösterir.

`hold` bağımsız değişkeni, sınama tamamlandıktan sonra dinleyicilerin çalışır durumda kalmasını sağlar.

Bu yapılandırma sırasında hatalar bildirilirse, yordamın aşağıdaki şekilde yeniden başlatılabilmesi için tüm değişiklikleri kaldırabilirsiniz:

1. Oluşturulan SOAP yerine JMS dizinini silin.
2. Kuyruk yöneticisini silin.

Bu görev hakkında

Bu örnek, tek yönlü kanal şekli kullanarak WebSphere MQ ' ta sağlanan JMS örnek hizmeti üzerinden .NET SOAP istemcisine bir WCF istemcisinden bir bağlantı gösterir. Hizmet, konsola bir metin dizesi çıkaran basit bir StockQuote örneği uygular.

The client has been generated by using WSDL to generate client files as described in “WSDL ile svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılanışı dosyaları oluşturuluyor” sayfa 589

Örnek, aşağıdaki yordamda açıklandığı gibi belirli kaynak adları ile yapılandırılmıştır. If you need to change the resource names, then you must also change the corresponding value on the client application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQNET\default\client\app.config file, and on the service application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQNET\default\service\WmqDefaultSample_StockQuoteDotNet.wsdl file, where *MQ_INSTALLATION_PATH* represents the installation directory for WebSphere MQ. JMS uç noktası URI 'sını biçimlendirme hakkında daha fazla bilgi için, WebSphere MQ ürün belgelerindeki *WebSphere MQ Transport for SOAP* başlıklı konuya bakın.

Yordam

İstemciyi bir kez çalıştırın: *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQNET\default\client\bin\Release\TestClient.exe dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , WebSphere MQkuruluş dizinini temsil eder.

İstemci uygulaması döngüleri beş kez örnek kuyruğa beş ileti gönderir.

Sonuçlar

Hizmet uygulaması, örnek kuyruktan iletileri alır ve ekranda Hello World beş kez görüntüler.

WebSphere MQ örneği tarafından barındırılan bir Axis Java hizmetine WCF istemcisi

Hem Java, hem de .NET için örnek istemci uygulamaları ve örnek hizmet yetkili sunucusu uygulamaları sağlanır. Örnekler, hisse senedi fiyat teklifi isteği alan bir hisse senedi senedi hizmetine dayalıdır ve hisse senedi alıntısını sağlar.

Başlamadan önce

This sample requires that the .NET SOAP over JMS service hosting environment is correctly installed and configured in WebSphere MQ and is accessible from a local queue manager. Ortamın kurulmasına ve yapılandırılmasına ilişkin bilgi için bkz. “SOAP için WebSphere MQ Web iletimi kurulması” sayfa 917

When the .NET SOAP over JMS service hosting environment is correctly installed and configured in WebSphere MQ and is accessible from a local queue manager, additional configuration steps must be completed.

1. WMQSOAP_HOME ortam değişkenini WebSphere MQ kuruluş dizinine (örneğin: C:\Program Files\IBM\WebSphere MQ) ayarlayın.
2. Ensure that the Java compiler javac is available and on the PATH.
3. axis.jar dosyasını, WebSphere kuruluş CD ' nin prereqs/axis dizininden WebSphere MQ kuruluş dizinine kopyalayın.
4. PATH değişkenine ekleyin: *MQ_INSTALLATION_PATH*\Java\lib burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini temsil eder; örneğin: C:\Program Files\IBM\WebSphere MQ
5. Ensure that the location of .NET is specified correctly in *MQ_INSTALLATION_PATH*\bin\amqwcallWSDL.cmd where *MQ_INSTALLATION_PATH* represents the directory where WebSphere MQ is installed, for example: C:\Program Files\IBM\WebSphere MQ. .NET ' in yeri örnek olarak belirtilebilir: set msfwdir=%ProgramFiles%\Microsoft Visual Studio .NET 2003\SDK\v1.1\Bin

Önceki adımlar tamamlanınca, hizmeti test edin ve çalıştırın:

1. JMS üzerinde çalışan JMS çalışma dizini için SOAP ' ınıza gidin.
2. Doğrulama sınamasını çalıştırmak ve hizmet dinleyicisini çalışır durumda bırakmak için aşağıdaki komutlardan birini girin:

- .NET için: *MQ_INSTALLATION_PATH*\Tools\soap\samples\runivt dotnet hold ; burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini temsil eder.
- AXIS:*MQ_INSTALLATION_PATH*\Tools\soap\samples\runivt Dotnet2AxisClient hold için, burada *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizini gösterir.

hold bağımsız değişkeni, sınama tamamlandıktan sonra dinleyicilerin çalışır durumda kalmasını sağlar.

Bu yapılandırma sırasında hatalar bildirilirse, yordamın aşağıdaki şekilde yeniden başlatılmasını için tüm değişiklikleri kaldırabilirsiniz:

1. Oluşturulan SOAP yerine JMS dizinini silin.
2. Kuyruk yöneticisini silin.

Bu görev hakkında

Örnek, bir WCF istemcisinden, tek yönlü kanal şeklini kullanarak WebSphere MQ ' da sağlanan JMS örnek hizmeti üzerinden Axis Java SOAP ile bağlantı sağlar. Hizmet, bir metin dizesini geçerli dizine kaydedilen bir dosyaya çıkaran basit bir StockQuote örneği uygular.

The client has been generated by using WSDL to generate client files as described in "WSDL ile svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılanışı dosyaları oluşturuluyor" sayfa 589

Örnek, bu paragrafta açıklandığı gibi, belirli kaynak adlarıyla yapılandırılmış olmalıdır. If you need to change the resource names, then you must also change the corresponding value on the client application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQAxis\default\client\app.config file, and on the service application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQAxis\default\service\WmqDefaultSample_StockQuoteDotNet.wsdl file, where *MQ_INSTALLATION_PATH* represents the installation directory for WebSphere MQ.

Yordam

İstemciyi bir kez çalıştırın: *MQ_INSTALLATION_PATH*\tools\wcf\samples\WMQAxis\default\client\bin\Release\TestClient.exe dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , WebSphere MQkuruluş dizinini temsil eder.

İstemci uygulaması döngüleri beş kez örnek kuyruğa beş ileti gönderir.

Sonuçlar

Hizmet uygulaması, örnek kuyruktan iletileri alır ve yürürlükteki dizinde bir dosyaya Hello World beş kez ekler.

İlgili başvurular

"Farklı SOAP yanıt öğesi adlarının işlenmesi" sayfa 603

WCF, döndürülen değerin adının varsayılan olarak belirli bir biçimde olmasını bekler, ancak bir hizmet, adı beklenen biçimde bir öğe döndürmeyebilir.

WebSphere Application Server örneğinin barındırdığı WCF istemcisi Java hizmetine

WebSphere Application Server (WAS) 6 için örnek istemci uygulamaları ve örnek hizmet yetkili sunucusu uygulamaları sağlanır. Bir istek-yanıt hizmeti de sağlanır.

Başlamadan önce

Bu örnek, aşağıdaki WebSphere MQ yapılanışlarının kullanılmasına gerek duyar:

Çizelge 77. WebSphere MQ için gerekli yapılandırma	
Nesne	**Gerekli ad**
Kuyruk yöneticisi	QM1
Yerel kuyruk	HelloWorld
Yerel kuyruk	HelloWorldYanıtla

Bu örnek ayrıca, bir WebSphere Application Server V6 barındırma ortamının doğru kurulup yapılandırılmasını gerektirir. WebSphere Application Server V6 , varsayılan olarak WebSphere MQ ' ya bağlanmak için bağ tanımlama kipi bağlantısı kullanır. Therefore WebSphere Application Server V6 must be installed on the same machine as the queue manager.

WAS ortamı yapılandırıldıktan sonra, aşağıdaki ek yapılandırma adımları tamamlanmalıdır:

1. WebSphere Application Server JNDI havuzunda aşağıdaki JNDI nesnelerini yaratın:
   a. `HelloWorld`adlı bir JMS kuyruğu hedefi
      - JNDI adını `jms/HelloWorld`olarak ayarlayın
      - Kuyruk adını `HelloWorld`olarak ayarlayın.
   b. A JMS queue connection factory called `HelloWorldQCF`
      - JNDI adını `jms/HelloWorldQCF`olarak ayarlayın
      - Kuyruk yöneticisi adını `QM1`olarak ayarlayın.
   c. A JMS queue connection factory called `WebServicesReplyQCF`
      - JNDI adını `jms/WebServicesReplyQCF`olarak ayarlayın
      - Kuyruk yöneticisi adını `QM1`olarak ayarlayın.
2. Aşağıdaki yapılanışı içeren WebSphere Application Server içinde `HelloWorldPort` adlı bir Message Listener kapısı yaratın:
   - Bağlantı üreticisi JNDI adını `jms/HelloWorldQCF`olarak ayarlayın.
   - Hedef JNDI adını `jms/HelloWorld`olarak ayarlayın
3. Web hizmeti `HelloWorldEJBEAR.ear` uygulamasını WebSphere Application Server sunucunuza aşağıdaki gibi kurun:
   a. **Uygulamalar** > **Yeni Uygulama** > **Yeni Kurumsal Uygulama**öğelerini tıklatın.
   b. *MQ_INSTALLATION_PATH*`\tools\wcf\samples\WAS\HelloWorldsEJBEAR.ear` 'a gidin; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' nun kuruluş dizinidir.
   c. Sihirbazdaki varsayılan seçeneklerden herhangi birini değiştirmeyin ve uygulama kurulduktan sonra uygulama sunucusunu yeniden başlatın.

WAS yapılandırması tamamlandığında, hizmeti bir kez çalıştırarak sınayarak sınayarak aşağıdakileri gerçekleştirin:

1. JMS üzerinde çalışma dizini üzerinde Soap 'ınıza gidin.
2. Örneği çalıştırmak için bu komutu girin: *MQ_INSTALLATION_PATH*`\tools\wcf\samples\WAS\TestClient.exe` ; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' nun kuruluş dizinidir.

Bu görev hakkında

The sample demonstrates a connection from a WCF client to the WebSphere Application Server SOAP over JMS sample service provided in the WCF samples included in WebSphere MQ V7, using a request-response channel shape. Messages flow between WCF and the WebSphere Application Server using WebSphere MQ queues. Hizmet, bir dizgiyi alan ve istemciye bir selamlama döndüren `HelloWorld(...)` yöntemini uygular.

"Çalışan bir hizmetteki meta verileri içeren svcutil aracını kullanarak bir WCF istemcisi yetkili sunucusu ve uygulama yapılandırma dosyaları oluşturulması" sayfa 588içinde açıklandığı gibi, ayrı olarak gösterilen bir HTTP uç noktasından hizmet meta verilerini almak için svcutil aracı kullanılarak istemci üretildi.

Örnek, aşağıdaki yordamda açıklandığı gibi belirli kaynak adları ile yapılandırılmıştır. If you need to change the resource names, then you must also change the corresponding value on the client application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WAS\default\client\app.config file, and on the service application in the *MQ_INSTALLATION_PATH*\tools\wcf\samples\WAS\HelloWorldsEJBEAR.ear where *MQ_INSTALLATION_PATH* is the installation directory of WebSphere MQ. JMS uç noktası URI 'sını biçimlendirme hakkında daha fazla bilgi için bkz. Web hizmeti konuşlandırması için URI sözdizimi ve parametreleri .

Hizmet ve istemci, IBM Geliştirici yazısı *JMS ve WebSphere Studio üzerinden SOAP kullanarak JMS Web hizmeti oluşturma*' da belirtilen hizmet ve istemciye dayanır. If you want to learn more about developing SOAP over JMS Web services which are compatible with the WebSphere MQ WCF custom channel, the relevant article can be found at: https://www.ibm.com/developerworks/websphere/library/techarticles/0402_du/0402_du.html.

Yordam

İstemciyi bir kez çalıştırın: *MQ_INSTALLATION_PATH*\tools\wcf\samples\WAS\default\client\bin\Release\TestClient.exe dosyasını çalıştırın; burada *MQ_INSTALLATION_PATH* , WebSphere MQ' un kuruluş dizinidir.

İstemci uygulaması hizmet yöntemlerinin her ikisini de aynı anda başlatır, örnek kuyruğa iki ileti gönderir.

Sonuçlar

Hizmet uygulaması, iletileri örnek kuyruktan alır ve istemci uygulaması çıkışlarının konsola verdiği HelloWorld(...) yöntemi çağrısına yanıt sağlar.

WebSphere MQiçin WCF özel kanalındaki sorun belirleme

WebSphere MQ kodunun çeşitli kısımlarıyla ilgili ayrıntılı bilgileri toplamak için WebSphere MQ izlemesini kullanabilirsiniz. Pencereler Communication Foundation (WCF) kullanılırken, WCF özel kanal izlemesi için Microsoft WCF altyapı izlemesiyle bütünleştirilmiş ayrı bir izleme çıkışı yaratılır.

WCF özel kanalına ilişkin izleme özelliğinin tam olarak etkinleştirilmesi iki çıkış dosyası üretir:

1. WCF özel kanal izlemesi, Microsoft WCF altyapı izlemesiyle tümleştirilmiştir.
2. WCF özel kanal izleme, XMS .NET ile tümleştirilmiş.

İki izleme çıktısına sahip olarak, her bir arabirimde sorunlar izlenebilir. Örneğin, uygun araçlar kullanılabilir. Örneğin:

- Uygun Microsoft araçları kullanılarak WCF sorun belirleme işlemi.
- WebSphere MQ MQI istemci sorunları XMS izleme biçimini kullanır.

To simplify trace enablement, the .NET 3 TraceSource and XMS .NET trace stack are both controlled using a single interface as described in: "WCF izleme yapılanışı ve izleme kütüğü adları" sayfa 598.

WCF özel kanal kural dışı durumu sıradüzeni

Özel kanal tarafından yayınlanan kural dışı durumlar tipleri WCF ile tutarlıdır ve tipik olarak bir TimeoutException ya da CommunicationException (ya da CommunicationExceptionalt sınıfı) olur.

Bağlantı ya da iç kural dışı durumlar kullanılarak, varsa, hata koşuluna ilişkin ek ayrıntılar sağlanır. Aşağıdaki istisnalar tipik örneklerdir ve kanalın mimarındaki her bir katman ek bağlantılı bir kural dışı duruma katkıda bulunur; örneğin; CommunicationsException , bağlantılı bir MQException içeren bir XMSException içeriyor:

1. System.serviceModel.CommunicationsExceptions

2. IBM.XMS.XMSException
3. IBM.WMQ.MQException

Anahtar bilgileri yakalanır ve sıradüzendeki en yüksek CommunicationException veri toplama işlemi sırasında sağlanır. Verilerin yakalanması ve sağlanması, uygulamaların bağlantılı kural dışı durumları sorgulamak ve içerebileceği ek bilgileri sorgulamak için kanalın mimarındaki her bir katmana bağlanmasını önlemektedir. Şu anahtar adları tanımlandi:

- IBM.XMS.WCF.ErrorCode: Yürürlükteki özel kanal kural dışı durumunun hata iletisi kodu.
- IBM.XMS.ErrorCode: Yığıdaki ilk XMS kural dışı durumunun hata iletisi.
- IBM.WMQ.ReasonCode: Temeldeki WebSphere MQ neden kodu.
- IBM.WMQ.CompletionCode: Temeldeki WebSphere MQ tamamlama kodu.

WCF izleme yapılanışı ve izleme kütüğü adları

İzleme tam olarak etkinleştirildiğinde, biri WCF sorunlarını tanılamak için bir tane olmak üzere iki çıkış dosyası ve iç izleme tanılama malzemesi için bir ayrıntılı dosya üretir. İzleme etkinleştirmesini kolaylaştırmak için hem .NET 3 TraceSource , hem de XMS .NET izleme yığınları tek bir arabirim kullanır.

WCF özel kanalı için iki farklı izleme yöntemi vardır; bu iki izleme yöntemi bağımsız olarak ya da birlikte etkinleştirilir. Her bir yöntem kendi izleme dosyasını üretir, bu nedenle her iki izleme yöntemi de etkinleştirildiğinde, iki izleme çıkış dosyası oluşturulur.

Yapılandırmayı ve etkinleştirmeyi mümkün olduğu kadar basit tutmak için, her iki izleme yöntemini de denetlemek için aynı arabirim kullanılır. `app.config` dosyasının, aşağıdaki bölümde anlatıldığı gibi ilgili izleme yapılandırmasını içerecek şekilde düzenlenmelidir. Böylece kullanıcılar, çıkışı kendi uygulamalarından izlemeyle birleştirmek için kendi eşdeğer bölümleri ekleyebilirler.

WCF özel kanal izleme varsayılan olarak etkinleştirilmedi. Önce bir izleme dinleyicisi yaratmalı, sonra `app.config` dosyasında seçilen izleme kaynağı için gereken izleme düzeyini ayarlayabilirsiniz.

WCF özel kanalının WCF altyapı izlemesiyle yapılandırılması

Aşağıdaki kod bölümünü `app.config` dosyasındaki `<system.diagnostics><sources>` bölümüne ekleyin:

```
<source name="IBM.XMS.WCF" switchValue="Verbose,ActivityTracing">
    <listeners>
        <remove name="Default"/>
        <add name="NewListener"/>
    </listeners>
</source>
```

Önceki kod parçası, .NET 3 TraceSource' u kullanarak kanal izlemesini yapar. Yürütülür dosyalarla ilişkili yapılanış kütüklerine ilişkin tüm çağrılar bu kod parçasıyla denetlenir.

WCF özel kanalının XMS .NET izlemesiyle yapılandırılması

Configuring the XMS .NET trace requires that you add a section of code to the `<system.diagnostics><sources>` section in the `app.config` file. Ancak, kod parçası, WCF özel kanalının WCF altyapı izlemesiyle yapılandırılması bölümünde gösterilen genişletilebilir `<source>` öğesine eklenir. Bu nedenle, XMS .NET izleme kodunun çalışması için WCF altyapısı izleme kodu var olsa da, WCF izlemesi etkinleştiriyor bölümünde açıklandığı gibi, WCF altyapısı izlemesi geçersiz kılınabilir.

```
<source name="IBM.XMS.WCF" switchValue="Verbose, ActivityTracing"
 xmsTraceSpecification="*=all=enabled" xmsTraceFilePath="path"
 xmsTraceFileSize="2000000" xmsTraceFileNumber="4" xmsTraceFormat="advanced">
    <listeners>
        <remove name="Default"/>
        <add name="NewListener"/>
    </listeners>
</source>
```

WCF izleme yapılanışı değişkenleri

Çizelge 78. WCF izleme yapılanışı değişkenleri

Değişken	**Tanım**
ad	Adı şu şekilde belirtin: `IBM.XMS.WCF`
switchValue	switchValue , izleme düzeyini denetler. When switchValue is set to `Kapalı`, the WCF infrastructure TraceSource is not generated. `Ayrıntılı`gibi başka bir değer de TraceSourceoluşturur. Microsoft' tan ayrıntılı izleme düzeyi bilgileri için WCF belgelerinize bakın ya da Microsoft WCF İzleme Web sayfasına gidin: https://msdn.microsoft.com/en-us/library/ms733025(vs.85).aspx

xmsTraceSpecification =*ComponentName=type=state*	*ComponentName* , izlenmesini istediğiniz sınıfın adıdır. Bu adda bir * genel arama karakteri kullanabilirsiniz. Örneğin:
	`*=all=enabled`
	Tüm sınıfları izlemek istediğinizi belirler ve
	`IBM.XMS.impl.*=all=enabled`
	Yalnızca API izlemesi gerektiğini belirtir.
	tip aşağıdaki izleme tiplerinden herhangi biri olabilir:
	• tümü
	• hata ayıklama
	• olay
	• EntryExit
	durum etkinleştirilebilir ya da devre dışı bırakılabilir.

xmsTraceFilePath= "*kütükadı*"	Bir xmsTraceFilePathbelirtmezseniz ya da xmsTraceFilePath varsa, ancak boş bir dizgi içeriyorsa, izleme dosyası yürürlükteki dizine yerleştirilir. İzleme dosyasını adlandırılmış bir dizinde saklamak için, xmsTraceFilePathiçinde şu dizin adını belirtin; örneğin: `xmsTraceFilePath="c:\somepath"`
xmsTraceFileSize= "*size*"	İzleme dosyası için izin verilen büyüklük üst sınırı. Bir dosya bu boyuta eriştiğinde arşivlenir ve yeniden adlandırılır. Varsayılan değer üst sınırı 20 KB 'dir. Bu değer, aşağıdaki şekilde belirlenir: `xmsTraceFileSize="20000000".`
xmsTraceFileNumber= "*sayı*"	Alıkonacağı izleme dosyalarının sayısı. Varsayılan değer 4 'tür (bir etkin dosya ve üç arşiv dosyası). İzin verilen alt sınır iki 'dir.

Çizelge 78. WCF izleme yapılanışı değişkenleri (devamı var)	
Değişken	**Tanım**

xmsTraceFormat="*biçim*"	İki düzey xmsTracebiçimi vardır: `basic` ve `advanced`. The default trace format is basic if you do not specify an xmsTraceFormat, or if the xmsTraceFormat is present but contains an empty string. Aşağıdaki özellikleri belirlerseniz, izleme dosyaları bu biçimde üretilir:
	`xmsTraceFormat="basic"`
	İzleme çözümleyici araçlarıyla uyumlu bir izleme gerektiriyorsa, şunları belirtmeniz gerekir:
	`traceFormat="advanced"`

WCF izlemesi etkinleştiriyor

İki farklı izleme yönteminin etkinleştirilmesi ve geçersiz kılınması için dört birleşim vardır. Dört birleşim, yukarıdaki bölümlerde açıklanan kodların bölümlerinin değerlerinin düzenlenmesini gerektirir.

Ayrıca, ayarlanabilen bir ortam değişkeni de vardır; daha fazla bilgi için bkz. “WCF_TRACE_ON ortam değişkeniyle WCF izlemesi etkinleştiriyor” sayfa 601.

Bu tablo ve gösterilen değerler, daha önce gösterilen kod parçalarına göre önceden `app.config` dosyasına eklenmiş olarak değişir.

Çizelge 79. WCF izleme etkinleştirme birleşimleri.		
İzleme tipi	**Değer değiştirildi**	**Örnek**
XMS izleme etkinleştirildi. WCF TraceSource etkinleştirildi	`switchValue` , `Off`(Kapalı) olarak ayarlanmamış	`<source name="IBM.XMS.WCF" switchValue="Verbose, ActivityTracing" xmsTraceSpecification="*=all=enabled" xmsTraceFilePath="path" xmsTraceFileSize="2000000" xmsTraceFileNumber="4" xmsTraceFormat="advanced"> <listeners> <remove name="Default"/> <add name="NewListener"/> </listeners> </source>`
XMS izleme etkinleştirildi. WCF TraceSource devre dışı	`switchValue` , `Off` (Kapalı) olarak ayarlanmıştır ve bir `xmsTraceSpecification` değeri verilmiştir.	`<source name="IBM.XMS.WCF" switchValue="Off, ActivityTracing" xmsTraceSpecification="*=all=enabled" xmsTraceFilePath="path" xmsTraceFileSize="2000000" xmsTraceFileNumber="4" xmsTraceFormat="advanced"> <listeners> <remove name="Default"/> <add name="NewListener"/> </listeners> </source>`

Çizelge 79. WCF izleme etkinleştirme birleşimleri. (devamı var)

İzleme tipi	Değer değiştirildi	Örnek
XMS izleme geçersiz kılındı. WCF TraceSource etkinleştirildi	Bu sonucu elde etmenin iki yolu vardır: • `switchValue` değişkeni `Off` (Kapalı) olarak ayarlanmamış ve bir`xmsTraceSpecification` değişkeni eklenmemiş. • `switchValue` değişkeni `Off` (Kapalı) olarak ayarlanmamış ve `xmsTraceSpecification` devre dışıolarak ayarlanmıştır.	`<source name="IBM.XMS.WCF" switchValue="Verbose, ActivityTracing" xmsTraceSpecification="*=all=disabled" xmsTraceFilePath="path" xmsTraceFileSize="2000000" xmsTraceFileNumber="4" xmsTraceFormat="advanced"> <listeners> <remove name="Default"/> <add name="NewListener"/> </listeners> </source>`
XMS izleme geçersiz kılındı. WCF TraceSource devre dışı	Bu sonucu elde etmenin üç yolu vardır: • `app.config` dosyasında `<source>` öğesi yok • `switchValue` değişkeni `Off` (Kapalı) olarak ayarlanmıştır ve bir`xmsTraceSpecification` değişkeni eklenmemiş • `switchValue` değişkeni `Off` (Kapalı) olarak ayarlanmıştır ve `xmsTraceSpecification` , `devre dışı`olarak ayarlanmıştır.	`<source name="IBM.XMS.WCF" switchValue="Off, ActivityTracing" xmsTraceSpecification="*=all=disabled" xmsTraceFilePath="path" xmsTraceFileSize="2000000" xmsTraceFileNumber="4" xmsTraceFormat="advanced"> <listeners> <remove name="Default"/> <add name="NewListener"/> </listeners> </source>`

WCF_TRACE_ON ortam değişkeniyle WCF izlemesi etkinleştiriyor

WCF izlemesini geçerli kılmak üzere tanımlanan önceki yöntemlerin yanı sıra, WCF_TRACE_ON ortam değişkeni kullanılarak XMS .NET izlemesi de etkinleştirilebilir.

WCF_TRACE_ON ortam değişkeninin boş değer olmayan herhangi bir değere ayarlanması, `xmstraceSpecification` ile `*=all=enable`darasındaki ayarın eşdeğeridir; örneğin: "`set WCF_TRACE_ON=true`"

Ancak, `xmstraceSpecification` dosyası `app.config` dosyasında açık bir şekilde ayarlandıysa, WCF_TRACE_ON ortam değişkeni geçersiz kılınır.

WCF izleme çıkış dosyaları ve dosya adları

XMS trace files are traditionally named using the base name and process ID format of: `xms_trace_`*`pid`*`.log`, where *pid* is the process ID.

As XMS trace files can still be produced in parallel with WCF custom channel trace files, the WCF custom channel trace integrated with XMS .NET trace output files have the following format to avoid confusion: `wcfxms_trace_`*`pid`*`.log`, where *pid* is the process ID.

İzleme çıkış dosyası varsayılan olarak yürürlükteki çalışma dizininde yaratılır, ancak gerekirse bu hedef yeniden tanımlanabilir.

WCF XMS İlk Hata Destek Teknolojisi (FFST)

WebSphere MQ kodunu kullanarak WebSphere MQ kodunun çeşitli bölümlerinin ne yaptığına ilişkin ayrıntılı bilgi toplayabilirsiniz. XMS FFST , WCF özel kanalına ilişkin kendi yapılandırma ve çıkış dosyalarını içerir.

XMS FFST izleme dosyaları geleneksel olarak, temel ad ve işlem tanıtıcısı biçimi kullanılarak adlandırılır: `xmsffdc`*`pid`*`_`*`date`*`.txt`; burada *pid* , işlem tanıtıcısıdır ve *tarih* saat ve tarihtir.

XMS FFT izleme dosyaları, WCF özel kanal XMS FFT dosyalarıyla paralel olarak üretilebildiğinden, WCF özel kanalı XMS FFT çıkış dosyaları karışıklığı önlemek için şu biçimlere sahiptir: `wcfffdc`*`pid`*`_`*`date`*`.txt`; burada *pid* , işlem tanıtıcısıdır ve *tarih* , saat ve tarihtir.

Bu izleme çıkış dosyası varsayılan olarak yürürlükteki çalışma dizininde yaratılır, ancak gerekirse bu hedef yeniden tanımlanabilir.

XMS .NET izleme üstbilgisiyle WCF özel kanalı aşağıdaki örneğe benzer:

```
************* Start Display XMS WCF Environment *************
  Product Name  :- value
  WCF Version   :- value
  Level    :- value
*************  End Display XMS WCF Environment  *************
```

FFST izleme dosyaları, özel kanala özgü herhangi bir biçimlendirme olmadan, standart şekilde biçimlendirilir.

WCF sürüm bilgileri

WCF sürüm bilgileri sorun saptamaya yardımcı olur ve özel kanala ilişkin derleme meta verilerinde bulunur.

WCF sürüm meta verileri için WebSphere MQ özel kanalı aşağıdaki üç yöntemden biriyle alınabilir:

- WebSphere MQ Utility dspmqver yardımcı programını kullanma. Dspmqver kullanımına ilişkin bilgi için bkz. dspmqver
- Windows Explorer özellikleri iletişim kutusunu kullanarak: Windows Explorer 'da **IBM.XMS.WCF.dll** > **Properties** > **Version**(Özellikler-> Sürüm).
- Herhangi bir kanaldan FFST ya da izleme dosyalarının üstbilgi bilgilerinden. FFST üstbilgi bilgisine ilişkin ek bilgi için bkz: “WCF XMS İlk Hata Destek Teknolojisi (FFST)” sayfa 601

WCF ipuçları ve ipuçları

Aşağıdaki ipuçları ve ipuçları önemli değildir ve belgelerin yeni sürümleri serbest bırakıldığında eklenecektir. Bunlar, yapmakta olduğunuz çalışmayla ilgiliyse, sizi zaman kazanmanızı sağlar.

WCF hizmeti anasistemindeki kural dışı durumlar dışsallaştırılıyor

WCF hizmet anasisteminin barındırdığı hizmetler için; hizmet, WCF internals ya da kanal yığınından atılan işlenmeyen kural dışı durumlar varsayılan olarak dışsallaştırılmaz. Bu kural dışı durumlarla ilgili bilgi almak için bir hata işleyici kaydedilmelidir.

Aşağıdaki kod, bir hizmetin özniteliği olarak uygulanabilen hata işleyici hizmeti davranışının tanımlanmasını sağlayan bir örnek sağlar:

```
using System.ServiceModel.Dispatcher;
using System.Collections.ObjectModel;
....
    public class ErrorHandlerBehaviorAttribute : Attribute, IServiceBehavior, IErrorHandler
    {
        //
        // IServiceBehavior Interface
        //
        public void AddBindingParameters(ServiceDescription serviceDescription,
           ServiceHostBase serviceHostBase, Collection<ServiceEndpoint> endpoints,
           BindingParameterCollection bindingParameters)
        {
        }
        public void ApplyDispatchBehavior(ServiceDescription serviceDescription,
            ServiceHostBase serviceHostBase)
        {
            foreach (ChannelDispatcher channelDispatcher in serviceHostBase.ChannelDispatchers)
{
```

```
                channelDispatcher.ErrorHandlers.Add(this);
            }
        }
        public void Validate(ServiceDescription serviceDescription, ServiceHostBase
serviceHostBase)
        {
        }

        //
        // IErrorHandler Interface
        //
        public bool HandleError(Exception e)
        {
            // Process the exception in the required way, in this case just outputting to the
console
            Console.Out.WriteLine(e);

            // Always return false to allow any other error handlers to run
            return false;
        }
        public void ProvideFault(Exception error, MessageVersion version, ref Message fault)
        {
        }
    }
```

Farklı SOAP yanıt öğesi adlarının işlenmesi

WCF, döndürülen değerin adının varsayılan olarak belirli bir biçimde olmasını bekler, ancak bir hizmet, adı beklenen biçimde bir öğe döndürmeyebilir.

WCF, döndürülen değerin şu biçimde adlandırılması için gereken kurala sahiptir: *methodName*`Result` ; burada *methodName* hizmet işleminin adıdır. For example, for a service called `getQuote`, WCF expects the response to be called: `getQuoteResult`.

Ancak, hizmet bu biçime uymayan bir adı taşıyan bir öğe döndürebilir.

Bir yetkili istemci oluşturmak için scvutill aracını çalıştırırken, WSDL farklı bir ad belirtiyorsa, yetkili arabirim, WCF ' ye arama yönergelerine arama yapmak için değiştirge ekler. Örneğin:

```
[System.ServiceModel.OperationContractAttribute(Action = "", ReplyAction = "*")]
[System.ServiceModel.XmlSerializerFormatAttribute(Style =
System.ServiceModel.OperationFormatStyle.Rpc,
                                                  Use =
System.ServiceModel.OperationFormatUse.Encoded)]
[return: System.ServiceModel.MessageParameterAttribute(Name = "getQuoteReturn")]
float getQuote(string in0);
```

Kendi arabiriminizi (örneğin, var olan bir yetkili sunucu arabirimine bir istek yanıt yöntemi ekleyerek) yaratayorsanız, hizmet farklı bir ad döndürürse, aynı parametreleri arabirime eklediğinizden emin olmalısınız. Bunu yapmazsanız, en sık rastlanan sorun, hizmet yöntemine yapılan bir çağrısının her zaman boş değer döndürmesi; bir nesne döndürülürse, yöntem boş değer döndürür, ancak tamsayı gibi bir sayısal değer döndürülürse, yöntem 0 değerini döndürür.

C++ kullanılması

WebSphere MQ provides C++ classes equivalent to WebSphere MQ objects and some additional classes equivalent to the array data types. Bu, MQI aracılığıyla olmayan bazı özellikleri sağlar.

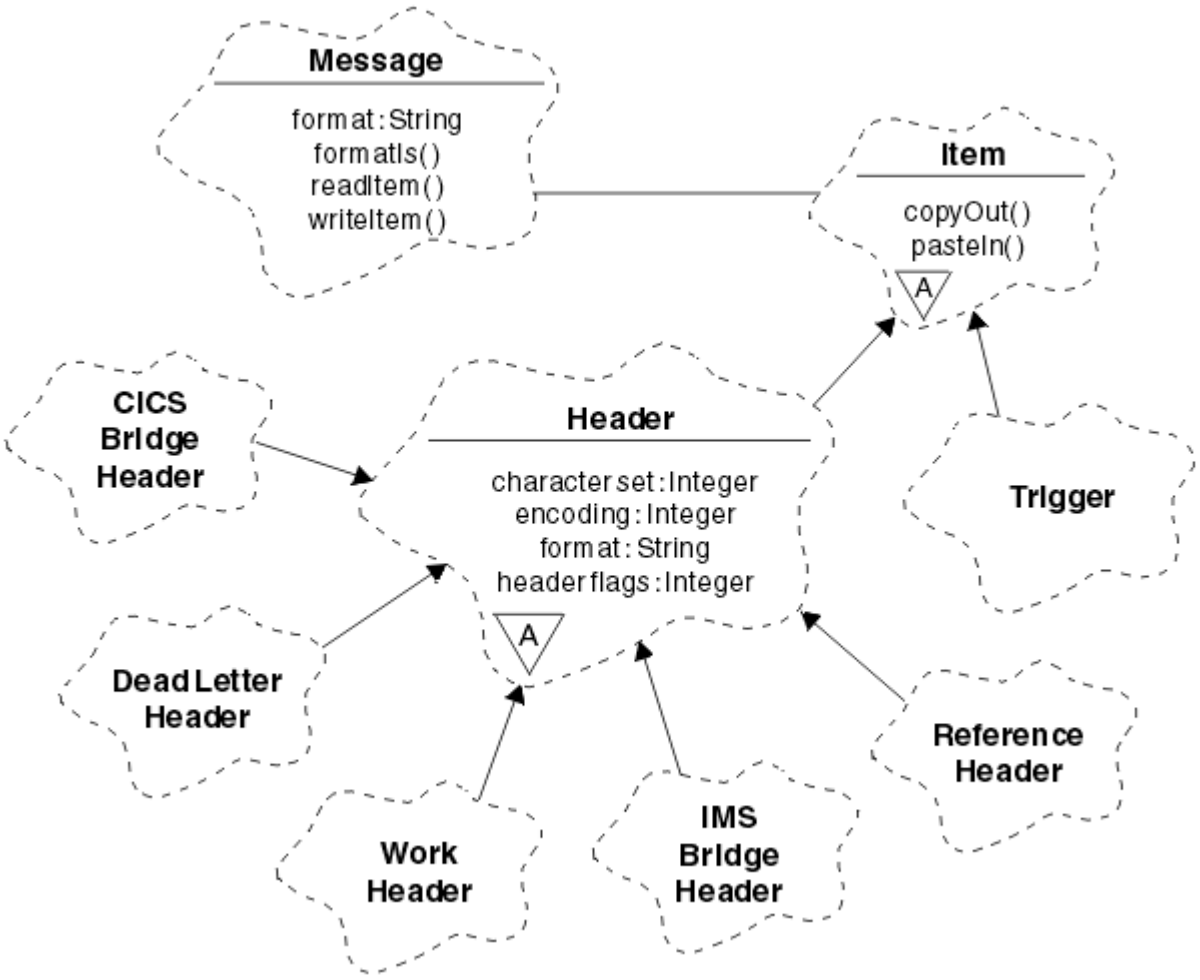
WebSphere MQ Sürüm 7.0itibariyle, WebSphere MQ programlama arabirimlerinde yapılan geliştirmeler C++ sınıflarına uygulanmaz.

WebSphere MQ C++ aşağıdaki özellikleri sağlar:

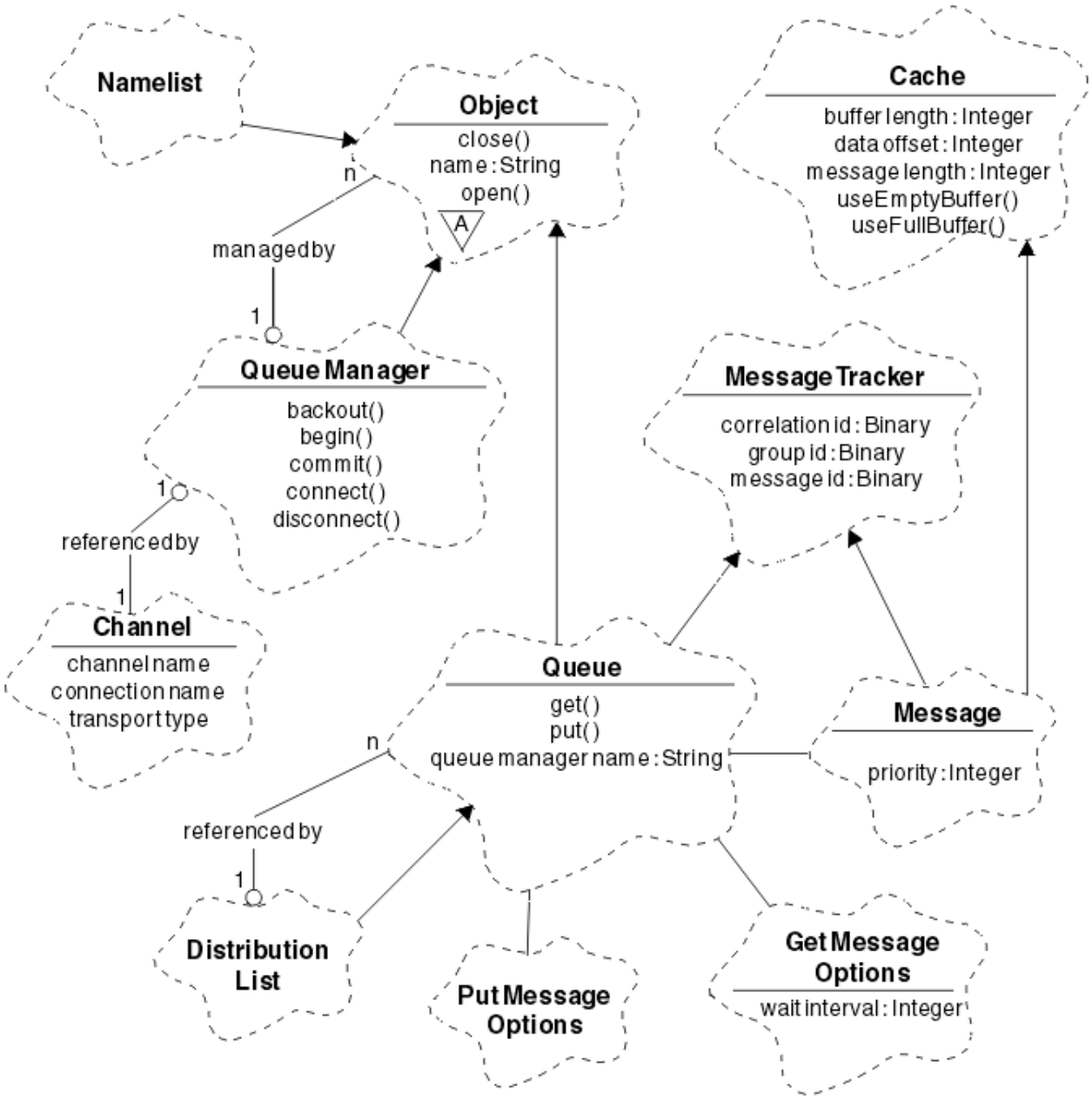
- WebSphere MQ veri yapılarının otomatik olarak başlatılması.
- Yalnızca zaman içi kuyruk yöneticisi bağlantısı ve kuyruk açma.
- Örtülü kuyruk kapatma ve kuyruk yöneticisi bağlantısı kesiyor.
- -Ölü harf başlığı iletimi ve makbumu.
- IMS köprü üstbilgisi iletimi ve girişi.

- İleti üstbilgisi iletimine ve girişine başvuruda bulunun.
- İleti girişini tetikle.
- CICS köprü üstbilgisi iletimi ve girişi.
- İş üstbilgisi iletimi ve girişi.
- İstemci kanalı tanımlaması.

The following Booch class diagrams show that all the classes are broadly parallel to those WebSphere MQ entities in the procedural MQI (for example using C) that have either handles or data structures. Tüm sınıflar ImqError sınıfından devralır (bkz. ImqError C++ sınıfı); bu, her nesneyle bir hata koşulunun ilişkilendirilmesini sağlar.



Şekil 120. WebSphere MQ C++ sınıfları (öğe işleme)



Şekil 121. WebSphere MQ C++ sınıfları (kuyruk yönetimi)

Booch sınıf çizgelerini doğru yorumlamak için aşağıdaki kurallardan haberdar olun:

- Yöntemler ve dikkate değer öznitelikler *sınıf* adının altında gösterilir.
- Bir bulut içindeki küçük bir üçgen *soyut sınıf* anlamına gelir.
- *Edinme* , üst sınıfa bir ok ile gösterilir.
- Bulutlar arasında süslenmemiş bir çizgi, sınıflar arasında bir *işbirliği ilişkisi* anlamına gelir.
- Sayı ile dekore edilen bir satır, iki sınıf arasındaki *gönderisel ilişki* anlamına gelir. Bu sayı, belirli bir ilişkiye herhangi bir anda katılabilecek nesnelerin sayısını gösterir.

Aşağıdaki sınıflar ve veri tipleri, kuyruk yönetimi sınıflarının C++ yöntemi imzalarında kullanılır (bkz. Şekil 121 sayfa 605) ve öğe işleme sınıflarına (bkz. Şekil 120 sayfa 604):

- ImqBinary sınıfı (bkz. ImqBinary C++ sınıfı), MQBYTE24gibi bayt dizilerini kapsüller.
- **typedef imzalanmamış char ImqBoolean**olarak tanımlanan ImqBoolean veri tipi.
- ImqString sınıfı (bkz. ImqString C++ sınıfı); MQCHAR64gibi karakter dizilerini sarmalayan bir sınıf.

Veri yapılarına sahip varlıklar, uygun nesne sınıfları içinde alt toplamlardır. Her bir veri yapısı alanına (bkz. C++ ve MQI çapraz başvuru) yöntemlerle erişilir.

Tutamaçları olan varlıklar ImqObject sınıf sıradüzeninin altında (bkz. ImqObject C++ sınıfı) ve MQI ' ya kapsüllenmiş arabirimler sağlar. Bu sınıfların nesneleri, yordamsal MQI ' ye göre gerekli olan yöntem çağrılarının sayısını azaltan akıllı davranış sergiler. Örneğin, kuyruk yöneticisi bağlantılarını gerektiği şekilde kurabilir ve atabilir ya da uygun seçeneklerle bir kuyruk açabilir, daha sonra bu bağlantıları kapatabilirsiniz.

The ImqMessage class (see ImqMessage C++ sınıfı) encapsulates the MQMD data structure and also acts as a holding point for user data and *öğe* (see “C++ dilinde ileti okunuyor” sayfa 615) by providing cached buffer facilities. Kullanıcı verileri için değişmez uzunluklu arabellekler sağlayabilir ve arabelleği birçok kez kullanabilirsiniz. Arabelleğindeki veri miktarı, bir kullanımdan diğerine farklılık gösterebilir. Diğer bir seçenek olarak, sistem, esnek uzunluktaki bir arabellek sağlayabilir ve bu arabelleği yönetebilir. Arabelleğin büyüklüğü (iletilerin alınması için kullanılabilir tutar) ve gerçekte kullanılan miktar (iletim için bayt sayısı ya da gerçekte alınan bayt sayısı) önemli noktalar haline gelir.

İlgili kavramlar

Teknik genel bakış

“C++ örnek programları” sayfa 606
İleti almayı ve yerleştirmeyi göstermek için dört örnek program sağlanır.

“C++ dilindeki önemli noktalar” sayfa 610
Bu konu derlemi, Message Queue Interface (MQI) olanağını kullanan uygulama programlarını yazarken göz önünde bulundurmanız gereken C++ dili kullanımı ve kurallarıyla ilgili konuları ayrıntılarıyla içerir.

“C++ dilinde ileti verileri hazırlanıyor” sayfa 614
İleti verileri, sistem ya da uygulama tarafından sağlanabilen bir arabelleğde hazırlanır. Her iki yöntemde de bazı avantajlar var. Bir arabelleğin kullanılmasına ilişkin örnekler verilmiştir.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“Uygulamaların geliştirilmesi” sayfa 7
IBM WebSphere MQ , iş süreçlerinizi desteklemek için gereksinim duyardığınız iletileri göndermek ve almak için uygulamalar geliştirebileceğiniz çeşitli yollar sağlar. Kuyruk yöneticilerinizi ve ilgili kaynaklarınızı yönetmek için de uygulamalar geliştirebilirsiniz.

İlgili başvurular

“Building WebSphere MQ C++ programs” sayfa 620
Desteklenen derleyicilerin URL 'si, C++ programlarını ve örneklerini WebSphere MQ altyapılarında derlemek, bağlamak ve çalıştırmak için kullanılacak komutlarla birlikte listelenir.

C++ ve MQI çapraz başvurusu

WebSphere MQ C++ sınıfları

C++ örnek programları

İleti almayı ve yerleştirmeyi göstermek için dört örnek program sağlanır.

Örnek programlar şunlardır:

- MERHABA DüNYANıN (imqwrld.cpp)
- SPUT (imqsput.cpp)
- SGET (imqsget.cpp)
- DPUT (imqdput.cpp)

Örnek programlar, Çizelge 80 sayfa 607içinde gösterilen dizinlerde bulunur.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Çizelge 80. Örnek programların yeri

Ortam	**Kaynağı içeren dizin**	**Yerleşik dizin programlar**
AIX	*MQ_INSTALLATION_PATH*/samp	*MQ_INSTALLATION_PATH*/samp/bin/ia
HP-UX	*MQ_INSTALLATION_PATH*/samp	*MQ_INSTALLATION_PATH*/ samp/bin/ah (bkz. not "2" sayfa 607)
Solaris	*MQ_INSTALLATION_PATH*/samp	*MQ_INSTALLATION_PATH*/samp/bin/as
Linux	*MQ_INSTALLATION_PATH*/samp	*MQ_INSTALLATION_PATH*/samp/bin/
Pencereler	*MQ_INSTALLATION_PATH*\tools\cplus\ samples	*MQ_INSTALLATION_PATH*\tools\cplus\ örnekler \bin\vn (bkz. not "3" sayfa 607)

Notlar:

1. Programs built using the ILE C++ compiler for IBM i are in the library QMQM. İçerme dosyaları / QIBM/ProdData/mqm/inc. içinde yer alır.
2. HP ANSI C++ derleyicisi kullanılarak oluşturulan programlar, *MQ_INSTALLATION_PATH*/ samp/bin/ah. Daha fazla bilgi için bkz. "C++ programlarını HP-UXüzerinde oluşturma" sayfa 621.
3. Microsoft Visual Studio kullanılarak oluşturulan programlar, *MQ_INSTALLATION_PATH*\tools\cplus\samples\bin\vn içinde bulunur. Bu derleyiciler hakkında daha fazla bilgi için bkz. "C++ programlarını Windowsüzerinde oluşturma" sayfa 627.

Örnek program HELLO WORLD (imqwrld.cpp)

Bu C++ örnek programı, ImqMessage sınıfını kullanarak düzenli bir veri paketi (C yapısı) nasıl yerleştireceğini ve nasıl alacağını gösterir.

Bu program, ImqMessage sınıfını kullanarak düzenli bir veri paketi (C yapısı) nasıl yerleştireceğini ve nasıl alacağını gösterir. Bu örnek, **aç**, **kapatma**ve **bağlantı kesme**gibi örtük yöntem çağrılarından yararlanmak için birkaç yöntem çağırımı kullanır.

z/OSdışındaki tüm platformlarda

WebSphere MQ' ya bir sunucu bağlantısı kullanıyorsanız, aşağıdaki yordamlardan birini izleyin:

- Varolan varsayılan kuyruğu kullanmak için SYSTEM.DEFAULT.LOCAL.QUEUE(Kuyruk), herhangi bir parametre geçirmeden **imqwrlds** programını çalıştırın
- Geçici olarak atanmış bir kuyruğu kullanmak için, **imqwrlds** komutunu kullanarak varsayılan model kuyruğunun adını (SYSTEM.DEFAULT.MODEL.QUEUE.

WebSphere MQ' ya istemci bağlantısı kullanıyorsanız, aşağıdaki yordamlardan birini izleyin:

- MQSERVER ortam değişkenini ayarlayın (ek bilgi için MQSERVER başlıklı konuya bakın) ve **imqwrldc**komutunu çalıştırın ya da
- Run **imqwrldc** passing as parameters the **queue-name**, **queue-manager-name**, and **channel-definition**, where a typical **channel-definition** might be SYSTEM.DEF.SVRCONN/TCP/*anasistem adı*(1414)

Örnek kod

```
extern "C" {
#include <stdio.h>
}

#include <imqi.hpp> // WebSphere MQ C++

#define EXISTING_QUEUE "SYSTEM.DEFAULT.LOCAL.QUEUE"

#define BUFFER_SIZE 12

static char gpszHello[ BUFFER_SIZE ] = "Hello world" ;
int main ( int argc, char * * argv ) {
  ImqQueueManager manager ;
  int iReturnCode = 0 ;

  // Connect to the queue manager.
  if ( argc > 2 ) {
    manager.setName( argv[ 2 ] );
  }
  if ( manager.connect( ) ) {
    ImqQueue * pqueue = new ImqQueue ;
    ImqMessage * pmsg = new ImqMessage ;

    // Identify the queue which will hold the message.
    pqueue -> setConnectionReference( manager );
    if ( argc > 1 ) {
      pqueue -> setName( argv[ 1 ] );


      // The named queue can be a model queue, which will result in
      // the creation of a temporary dynamic queue, which will be
      // destroyed as soon as it is closed. Therefore we must ensure
      // that such a queue is not automatically closed and reopened.
      // We do this by setting open options which will avoid the need
      // for closure and reopening.
      pqueue -> setOpenOptions( MQOO_OUTPUT | MQOO_INPUT_SHARED |
                                MQOO_INQUIRE );
    } else {
      pqueue -> setName( EXISTING_QUEUE );

      // The existing queue is not a model queue, and will not be
      // destroyed by automatic closure and reopening. Therefore we
      // will let the open options be selected on an as-needed basis.
      // The queue will be opened implicitly with an output option
      // during the "put", and then implicitly closed and reopened
      // with the addition of an input option during the "get".
    }

    // Prepare a message containing the text "Hello world".
    pmsg -> useFullBuffer( gpszHello , BUFFER_SIZE );
    pmsg -> setFormat( MQFMT_STRING );

    // Place the message on the queue, using default put message
    // Options.
    // The queue will be automatically opened with an output option.
    if ( pqueue -> put( * pmsg ) ) {
      ImqString strQueue( pqueue -> name( ) );

      // Discover the name of the queue manager.
      ImqString strQueueManagerName( manager.name( ) );
      printf( "The queue manager name is %s.\n",
              (char *)strQueueManagerName );

      // Show the name of the queue.
      printf( "Message sent to %s.\n", (char *)strQueue );

      // Retrieve the data message just sent ("Hello world" expected)
      // from the queue, using default get message options. The queue
      // is automatically closed and reopened with an input option
      // if it is not already open with an input option. We get the
      // message just sent, rather than any other message on the
      // queue, because the "put" will have set the ID of the message
      // so, as we are using the same message object, the message ID
      // acts as in the message object, a filter which says that we
      // are interested in a message only if it has this
      // particular ID.

      if ( pqueue -> get( * pmsg ) ) {
```

```
        int iDataLength = pmsg -> dataLength( );

        // Show the text of the received message.
        printf( "Message of length %d received, ", iDataLength );

        if ( pmsg -> formatIs( MQFMT_STRING ) ) {
          char * pszText = pmsg -> bufferPointer( );

          // If the last character of data is a null, then we can
          // assume that the data can be interpreted as a text
          // string.
          if ( ! pszText[ iDataLength - 1 ] ) {
            printf( "text is \"%s\".\n", pszText );
          } else {
            printf( "no text.\n" );
          }

        } else {
          printf( "non-text message.\n" );
        }
      } else {
        printf( "ImqQueue::get failed with reason code %ld\n",
                pqueue -> reasonCode( ) );
        iReturnCode = (int)pqueue -> reasonCode( );
      }

    } else {
      printf( "ImqQueue::open/put failed with reason code %ld\n",
              pqueue -> reasonCode( ) );
      iReturnCode = (int)pqueue -> reasonCode( );
    }

    // Deletion of the queue will ensure that it is closed.
    // If the queue is dynamic then it will also be destroyed.
    delete pqueue ;
    delete pmsg ;

  } else {
    printf( "ImqQueueManager::connect failed with reason code %ld\n"
            manager.reasonCode( ) );
    iReturnCode = (int)manager.reasonCode( );
  }

  // Destruction of the queue manager ensures that it is
  // disconnected.  If the queue object were still available
  // and open (which it is not), the queue would be closed
  // prior to disconnection.

  return iReturnCode ;
}
```

Örnek programlar SPUT (imqsput.cpp) ve SGET (imqsget.cpp)

Bu C++ programları, adlandırılmış bir kuyruktan iletileri alır ve bu kuyruktan iletileri alır.

Bu örnekler, aşağıdaki sınıfların kullanımını gösterir:

- ImqError (bkz. ImqError C++ sınıfı)
- ImqMessage (bkz. ImqMessage C++ sınıfı)
- ImqObject (bkz. ImqObject C++ sınıfı)
- ImqQueue (bkz. ImqQueue C++ sınıfı)
- ImqQueueManager (bkz. ImqQueueManager C++ sınıfı)

Programları çalıştırmak için uygun yönergeleri izleyin.

z/OSdışındaki tüm platformlarda

1. Run **imqsputs** *kuyruk-adı*.
2. Konsolda metin satırları yazın. Bu satırlar, belirlenen kuyruğa ileti olarak yerleştirilir.
3. Girişi sonlamak için boş bir satır girin.

4. Tüm satırları almak ve bunları konsolda görüntülemek için **imqsgets** *kuyruk-adı* komutunu çalıştırın.

Örnek program DPUT (imqdput.cpp)

Bu C++ örnek programı, iletileri iki kuyruktan oluşan bir dağıtım listesine koyar.

DPUT, ImqDistributionListe sınıfının kullanımını gösterir (bkz. ImqDistributionList C++ sınıfı). Bu örnek, z/OSüzerinde desteklenmez.

1. İki adlandırılmış kuyruğa ileti yerleştirmek için **imqdlets** *queue-name-1 queue-name-2* komutunu çalıştırın.
2. Bu kuyruklardan gelen iletileri almak için **imqsgets** *queue-name-1* ve **imqsgets** *queue-name-2* komutunu çalıştırın.

C++ dilindeki önemli noktalar

Bu konu derlemi, Message Queue Interface (MQI) olanağını kullanan uygulama programlarını yazarken göz önünde bulundurmanız gereken C++ dili kullanımı ve kurallarıyla ilgili konuları ayrıntılarıyla içerir.

C++ Üstbilgi dosyaları

Üstbilgi kütükleri C++ dilinde WebSphere MQ uygulama programlarını yazmanıza yardımcı olmak için, MQI tanımının bir parçası olarak sağlanır.

Bu üstbilgi dosyaları aşağıdaki tabloda özetlenmiştir.

Çizelge 81. C/C++ üstbilgi kütükleri

Dosya adı	**İçindekiler**
IMQI.HPP	C++ MQI Sınıfları (CMQC.H ve IMQTYPE.H)
IMQTYPE.H	**ImqBoolean** veri tipini tanımlar
CMQC.H	MQI veri yapıları ve bildirge değişmezleri

Uygulamaların taşınabilirliğini artırmak için, **#include** ön işlemcisi yönergesinde üstbilgi dosyasının adını küçük harfli olarak kodlayın:

```
#include <imqi.hpp> // C++ classes
```

C++ yöntemleri ve öznitelikleri

Yöntem adları karışık durumda. Parametrelere ve dönüş değerlerine ilişkin çeşitli noktalar geçerlidir. Öznitelikler, set kullanılarak erişilir ve yöntemleri uygun olarak alır.

const olan yöntemlerin parametreleri yalnızca giriş içindir. Bir işaretçi (*) ya da başvuru (&) de içinde olmak üzere imzaları olan parametreler referans olarak iletilir. Bir işaretçi ya da başvuru içermeyen dönüş değerleri, değer temelinde geçirilir; döndürülen nesneler durumunda bunlar, çağıranın sorumluluğu olan yeni varlıklardır.

Bazı yöntem imzaları, belirtilmediyse, varsayılan olarak kabul edilen öğeleri içerir. Bu tür öğeler her zaman imzaların sonunda olur ve eşittir işaretiyle (=) belirtilir; eşittir işareti, öğe atılırsa geçerli olan varsayılan değeri belirtir.

Bu sınıflardaki tüm yöntem adları küçük harfle başlayarak büyük ve küçük harfe karıştırılır. Bir yöntem adı içindeki ilk dışında her sözcük, büyük harfle başlar. Kısaltmaların anlamı geniş bir şekilde anlaşılmadıkça kullanılmaz. Kullanılan kısaltmalar arasında *tanıtıcı* (kimlik için) ve *eşitleme* (eşitleme için) vardır.

Nesne özniteliklerine, set ve get yöntemleri kullanılarak erişilir. Bir set yöntemi *set*sözcüğüyle başlar; get method has no prefix. Bir öznitelik *salt okunur*ise, herhangi bir ayarlama yöntemi yoktur.

Öznitelikler, nesne yapısı sırasında geçerli durumlarla başlatılır ve bir nesnenin durumu her zaman tutarlıdır.

C++ dilinde veri tipleri

Tüm veri tipleri C **typedef** deyimiyle tanımlanır.

ImqBoolean tipi, IMQTYPE.H içinde **işaretsiz karakter** olarak tanımlanır ve TRUE ve FALSE değerlerine sahip olabilir. You can use **ImqBinary** class objects in place of **MQBYTE** arrays, and **ImqString** class objects in place of **char ***. Çoğu yöntem, depolama yönetimini kolaylaştırmak için **char** ya da **MQBYTE** işaretçileri yerine nesneleri döndürür. Tüm dönüş değerleri çağıranın sorumluluğu haline gelir ve döndürülen bir nesne durumunda, saklama alanı silme işlemi kullanılarak atılabilir.

C++ dilinde ikili dizgileri işleme

İkili veriler dizgileri, **ImqBinary** sınıfının nesnesi olarak bildirilir. Bu sınıfın nesneleri, bilinen C işleçlerini kullanarak kopyalanabilir, karşılaştırılabilir ve bu işleçler kullanılarak ayarlanabilirler. Örnek kod sağlanıyor.

Aşağıdaki kod örneği, ikili bir dizilimin üzerindeki işlemleri gösterir:

```
#include <imqi.hpp> // C++ classes

ImqMessage message ;
ImqBinary id, correlationId ;
MQBYTE24 byteId ;

correlationId.set( byteId, sizeof( byteId ) ); // Set.
id = message.id( );                            // Assign.
if ( correlationId == id ) {                   // Compare.
  ...
```

C++ dilinde karakter dizilimlerini işleme

Karakter verileri genellikle, bir dönüştürme işleci kullanılarak **char *** tipine dönüştürülebilen **ImqString** sınıf nesnelerinde döndürülür. ImqString sınıfı, karakter dizgilerinin işlenmesine yardımcı olacak yöntemler içerir.

Karakter verileri kabul edildiğinde ya da MQI C++ yöntemleri kullanılarak döndürüldüğünde, karakter verileri her zaman boş değerle sonlandırılır ve herhangi bir uzunluğa sahip olabilir. Ancak, belirli sınırlar WebSphere MQ tarafından uygulanarak, bilgilerin kesilmesiyle sonuçlanabilir. Depolama yönetimini kolaylaştırmak için, karakter verileri genellikle **ImqString** sınıf nesnelerinde döndürülür. Bu nesneler, sağlanan dönüştürme işlecini kullanarak **char *** 'a dönüştürülebilirler ve **char *** ' un gerekli olduğu birçok durumda *salt okunur* için kullanılır.

Not: Bir **ImqString** sınıf nesnesinden **char *** dönüştürme sonucu boş değerli olabilir.

Although C functions can be used on the **char ***, there are special methods of the **ImqString** class that are preferable; **işletmen uzunluğu**( ) is the equivalent of **strlen** and **depolama**( ) indicates the memory allocated for the character data.

C++ dilinde nesnelerin ilk durumu

Tüm nesneler, özniteliklerine yansıtılan tutarlı bir başlangıç durumuna sahiptir. İlk değerler sınıf tanımlarında tanımlanır.

C++ ' den C++ Kullanılması

C işlevlerini bir C++ programından kullandığınızda, uygun üstbilgileri içerin.

Aşağıdaki örnek, bir C++ programına dahil edilen `string.h` ' i göstermektedir:

```
extern "C" {
```

```
#include <string.h>
}
```

C++ notasyonlu kuralları

Bu örnekte, yöntemlerin nasıl çağrılacağı ve parametrelerin bildirileceği gösterilmektedir.

Bu kod örneği şu yöntemleri ve değiştirgeleri kullanır:**ImqBoolean ImqQueue**::**get**(**ImqMessage &** *msg*)

Değiştirgeleri şu şekilde bildirin ve kullanın:

```
ImqQueueManager * pmanager ;    // Queue manager
ImqQueue * pqueue ;             // Message queue
ImqMessage msg ;                // Message
char szBuffer[ 100 ];          // Buffer for message data

pmanager = new ImqQueueManager ;
pqueue = new ImqQueue ;
pqueue -> setName( "myreplyq" );
pqueue -> setConnectionReference( pmanager );

msg.useEmptyBuffer( szBuffer, sizeof( szBuffer ) );

if ( pqueue -> get( msg ) ) {
  long lDataLength = msg.dataLength( );

  ...
}
```

C++ dilinde örtük işlemler

Bir yöntemin başarıyla yürütülmesine ilişkin önkoşul koşullarını yerine getirmek için, birden çok işlem örtük olarak *tam zamanında*ile oluşabilir. Bu örtülü işlemler, bağlantı, açma, yeniden açma, kapatma ve bağlantı kesme işlemleridir. Sınıf özniteliklerini kullanarak, bağlantı denetimi ve açık örtük davranışı denetleyebilirsiniz.

Bağlan

Bir ImqQueueManager nesnesi, MQI ' ya yönelik herhangi bir çağrıyla sonuçlanan herhangi bir yöntem için otomatik olarak bağlanır (bkz. C++ ve MQI çapraz başvurusu).

Au00e7

Bir MQGET, MQINQ, MQPUT ya da MQSET çağrısına neden olan herhangi bir yöntem için otomatik olarak bir ImqObject nesnesi açılır. Bir ya da daha fazla ilgili **açma seçeneği** değeri belirtmek için **openFor** yöntemini kullanın.

Yeniden aç

ImqObject , nesnenin zaten açık olduğu bir MQGET, MQINQ, MQPUT ya da MQSET çağrısına neden olan herhangi bir yöntem için otomatik olarak yeniden açılır; ancak, var olan **açık seçenekler** , MQI çağrısının başarılı olmasına izin vermek için yeterli değildir. Bu nesne, geçici olarak MQCO_NONE için geçici bir **kapatma seçenekleri** değeri kullanılarak kapatıldı. İlgili bir öğe eklemek için **openFor** yöntemini kullanın**seçeneği açın**.

Yeniden açma, belirli durumlarda sorunlara neden olabilir:

- Geçici bir dinamik kuyruk kapatıldığında yok edilir ve hiçbir zaman yeniden açılamayabilir.
- Dışlayıcı giriş (belirtik olarak ya da varsayılan olarak) için açılan bir kuyrukta, kapatma ve yeniden açma işlemi sırasında fırsat penceresindeki diğer kişiler tarafından erişilebilir.
- Bir kuyruk kapatıldığında göz atma konumu kaybedilir. Bu durum kapanmayı ve yeniden açmayı önlemez, ancak MQGMO_BROWSE_FIRST komutu yeniden kullanılmadıkça, imlecin sonraki kullanımını önler.

- Bir kuyruk kapatıldığında, alınan son iletinin bağlamı kaybedilir.

Bu koşullardan herhangi biri ortaya çıkarsa ya da öngörülebiliyorsa, bir nesne açılmadan önce (belirtik ya da örtük olarak) yeterli **açık seçenekleri** belirtik olarak ayarlayarak yeniden açılamamayı önleyebilirsiniz.

Karmaşık kuyruk işleme durumları için açık olarak **açık seçenekler** ' in ayarlanması daha iyi performans sağlar ve yeniden açma kullanımıyla ilişkili sorunları önler.

Kapat
ImqObject , nesne durumunun artık geçerli olmadığı herhangi bir noktada otomatik olarak kapatılır; örneğin, bir ImqObject bağlantı başvurusu kesilirse ya da bir ImqObject nesnesi yok edilmişse.

Bağlantıyı kes
ImqQueueYöneticisi, bağlantının artık geçerli olmadığı herhangi bir noktada otomatik olarak kesilir; örneğin, bir ImqObject bağlantı başvurusu kesilirse ya da bir ImqQueueManager nesnesi yok edildiyse.

C++ dilinde ikili ve karakter dizilimleri

ImqString sınıfı, geleneksel *char *** veri biçimini sarsalıyor. ImqBinary sınıfı, ikili bayt dizisini sarsalıyor. Karakter verilerini ayarlayan bazı yöntemler verilerin kesilmesine neden olabilir.

Karakter (**char ***) verilerini ayarlayan yöntemler her zaman verilerin bir kopyasını alır, ancak bazı yöntemler WebSphere MQtarafından uygulandığından, bazı yöntemler kopyayı kesebilir.

The ImqString class (see ImqString C++ sınıfı) encapsulates the traditional **char *** and provides support for:

- Karşılaştırma
- Bitiştirme
- Kopyalama
- Integer-to-text ve text-to-integer dönüştürme
- Simge (sözcük) çıkartma
- Büyük harf çevirisi

ImqBinary sınıfı (bkz. ImqBinary C++ sınıfı) rasgele boyutların ikili bayt dizilerini sarmalıyor. Özellikle, aşağıdaki öznitelikleri tutmak için kullanılır:

- **muhasebe simgesi** (MQBYTE32)
- **bağlantı etiketi** (MQBYTE128)
- **İlinti tanıtıcısı** (MQBYTE24)
- **tesis simgesi** (MQBYTE8)
- **grup tanıtıcısı** (MQBYTE24)
- **eşgörünüm tanıtıcısı** (MQBYTE24)
- **ileti tanıtıcısı** (MQBYTE24)
- **ileti simgesi** (MQBYTE16)
- **işlem eşgörünümü tanıtıcısı** (MQBYTE16)

Bu özniteliklerin bulunduğu yer, aşağıdaki sınıfların nesnelerine aittir:

- ImqCICSBridgeÜstbilgisi (bkz. ImqCICSBridgeHeader C++ sınıfı)
- ImqGetMessageOptions (bkz. ImqGetMessageOptions C++ sınıfı)
- ImqIMSBridgeÜstbilgisi (bkz. ImqIMSBridgeHeader C++ sınıfı)
- ImqMessageİzleyici (bkz. ImqMessageTracker C++ sınıfı)
- ImqQueueManager (bkz. ImqQueueManager C++ sınıfı)
- ImqReferenceÜstbilgisi (bkz. ImqReferenceHeader C++ sınıfı)

- ImqWorkÜstbilgisi (bkz. ImqWorkHeader C++ sınıfı)

ImqBinary sınıfı, karşılaştırma ve kopyalama desteği de sağlar.

C++ dilinde desteklenmeyen işlevler

WebSphere MQ C++ sınıfları ve yöntemleri, WebSphere MQ altyapısından bağımsızdır. Bu nedenle, bazı platformlarda desteklenmeyen bazı işlevler sunabilirler.

If you try to use a function on a platform on which it is not supported, the function is detected by WebSphere MQ but not by the C++ language bindings. WebSphere MQ , diğer herhangi bir MQI hatası gibi, hatayı programınıza bildirir.

C++ dilinde ileti alışverişi

Bu konular, C + + içinde iletilerin nasıl hazırlanacağını, okunacağını ve yazılacağını ayrıntılarıyla içerir.

C++ dilinde ileti verileri hazırlanıyor

İleti verileri, sistem ya da uygulama tarafından sağlanabilen bir arabelleğde hazırlanır. Her iki yöntemde de bazı avantajlar var. Bir arabelleğin kullanılmasına ilişkin örnekler verilmiştir.

Bir ileti gönderdiğinizde, ileti verileri ilk olarak bir ImqCache nesnesi tarafından yönetilen bir arabelleğde hazırlanır (bkz. ImqCache C++ sınıfı). Bir arabellek her ImqMessage nesnesiyle ilişkilendirilmiş (kalıtım temelinde) (bkz. ImqMessage C++ sınıfı): uygulama tarafından (**useEmptyBuffer** ya da **useFullBuffer** yöntemi kullanılarak) ya da sistem tarafından otomatik olarak sağlanabilir. İleti arabelleğini sağlayan uygulamanın yararı, uygulamanın hazırlanmış veri alanlarını doğrudan kullanabileceği için, birçok durumda veri kopyalamanın gerekli olmamasını sağlar. Bu dezavantaj, belirtilen arabelleğin değişmez uzunlukta olması.

Arabellek yeniden kullanılabilir ve iletilmeden önce **setMessageLength** yöntemi kullanılarak iletilen bayt sayısı her seferinde kullanılabilir kılınabilir.

Sistem tarafından otomatik olarak sağlandığında, kullanılabilir bayt sayısı sistem tarafından yönetilir ve veriler, örneğin ImqCache **write** yöntemi ya da ImqMessage **writeItem** yöntemi kullanılarak ileti arabelleğiyle kopyalanabilir. İleti arabelleği gereksinmeye göre büyür. Arabellek büyüdükçe, önceden yazılan veri kaybı olmaz. Büyük ya da çok parçalı bir ileti sıralı parçalarda yazılabilir.

Aşağıdaki örneklerde, basitleştirilmiş ileti gönderileri gösterilmektedir.

1. Kullanıcı tarafından sağlanan bir arabelleğde hazırlanmış verileri kullan

```
char szBuffer[ ] = "Hello world" ;

msg.useFullBuffer( szBuffer, sizeof( szBuffer ) );
msg.setFormat( MQFMT_STRING );
```

2. Kullanıcı tarafından sağlanan bir arabellete, arabellek büyüklüğünün veri büyüklüğünü aştığı verileri kullanın.

```
char szBuffer[ 24 ] = "Hello world" ;

msg.useEmptyBuffer( szBuffer, sizeof( szBuffer ) );
msg.setFormat( MQFMT_STRING );
msg.setMessageLength( 12 );
```

3. Verileri kullanıcı tarafından sağlanan bir arabelleğe kopyala

```
char szBuffer[ 12 ];

msg.useEmptyBuffer( szBuffer, sizeof( szBuffer ) );
```

```
msg.setFormat( MQFMT_STRING );
msg.write( 12, "Hello world" );
```

4. Sistem tarafından sağlanan bir arabelleğe veri kopyalama

```
msg.setFormat( MQFMT_STRING );
msg.write( 12, "Hello world" );
```

5. Nesneleri sistem tarafından sağlanan arabelleğe nesneleri kullanarak kopyalama (ileti biçimini ve içeriğin yanı sıra içerik)

```
ImqString strText( "Hello world" );

msg.writeItem( strText );
```

C++ dilinde ileti okunuyor

Arabellek, uygulama ya da sistem tarafından sağlanabilir. Verilere doğrudan arabellekten erişilebilir ya da sırayla okunabilirler. Her ileti tipine eşdeğer bir sınıf vardır. Örnek kod verilmiştir.

Veri alınırken, uygulama ya da sistem uygun bir ileti arabelleği sağlayabilir. Aynı arabellek, hem birden çok iletim için, hem de belirli bir ImqMessage nesnesi için birden çok giriş için kullanılabilir. İleti arabelleği otomatik olarak sağlanırsa, alınan veri uzunluğunu sığdırmak için büyür. Ancak, uygulama tarafından sağlanan bir ileti arabelleği, alınan verileri tutmak için yeterli olmayabilir. Daha sonra, ileti girişi için kullanılan seçeneklere bağlı olarak kesilme ya da hata oluşabilir.

Gelen verilere doğrudan ileti arabelleğinden erişilebilir; bu durumda, veri uzunluğu gelen verilerin toplam miktarını gösterir. Diğer bir seçenek olarak, gelen veriler ileti arabelleğinden sıralı olarak okunabilmektedir. Bu durumda, veri göstergesi gelen verilerin bir sonraki baytı adreslenir ve veri göstergesi ve veri uzunluğu her okunaca güncellenmektedir.

Öğeler , sırayla ve ayrı olarak işlenmesi gereken, ileti arabelleğindeki tüm kullanıcı alanında bulunan bir iletinin parçalarıdır. Normal kullanıcı verileri dışında, bir öğe ölü harf üstbilgisi ya da tetikleme iletisi olabilir. Öğeler her zaman ileti biçimleriyle ilişkilendirilir; ileti biçimleri ***değil*** her zaman öğelerle ilişkilendirilir.

Tanınabilir bir WebSphere MQ ileti biçimine karşılık gelen her öğe için bir nesne sınıfı vardır. Bir tane ölü harf üstbilgisi ve tetikleyici bir mesaj için bir tane var. Kullanıcı verileri için bir nesne sınıfı yok. Yani, tanınabilir biçimler tükendikten sonra, kalan kısmı işleme uygulama programına bırakılır. Kullanıcı verilerine ilişkin sınıflar, ImqItem sınıfı belirtilerek yazılabilir.

Aşağıdaki örnekte, kullanıcı verilerinden önce hayali bir durumda olabilecek bir dizi olası öğe dikkate alan bir ileti alındı örneği gösterilmektedir. Öğe olmayan kullanıcı verileri, tanımlanabilen öğeler sonrasında ortaya çıkan her şey olarak tanımlanır. Otomatik arabellek (varsayılan değer), ileti verilerinin rasgele bir miktarını tutmak için kullanılır.

```
ImqQueue queue ;
ImqMessage msg ;

if ( queue.get( msg ) ) {

  /* Process all items of data in the message buffer. */
  do while ( msg.dataLength( ) ) {
    ImqBoolean bFormatKnown = FALSE ;
    /* There remains unprocessed data in the message buffer. */

    /* Determine what kind of item is next. */

    if ( msg.formatIs( MQFMT_DEAD_LETTER_HEADER ) ) {
      ImqDeadLetterHeader header ;
      /* The next item is a dead-letter header.                */
```

```
      /* For the next statement to work and return TRUE,       */
      /* the correct class of object pointer must be supplied. */
      bFormatKnown = TRUE ;

      if ( msg.readItem( header ) ) {
        /* The dead-letter header has been extricated from the */
        /* buffer and transformed into a dead-letter object.   */
        /* The encoding and character set of the dead-letter   */
        /* object itself are MQENC_NATIVE and MQCCSI_Q_MGR.    */
        /* The encoding and character set from the dead-letter */
        /* header have been copied to the message attributes   */
        /* to reflect any remaining data in the buffer.        */

        /* Process the information in the dead-letter object.  */
        /* Note that the encoding and character set have       */
        /* already been processed.                             */
        ...
      }
      /* There might be another item after this, */
      /* or just the user data.                  */
    }
    if ( msg.formatIs( MQFMT_TRIGGER ) ) {
      ImqTrigger trigger ;
      /* The next item is a trigger message.                   */
      /* For the next statement to work and return TRUE,       */
      /* the correct class of object pointer must be supplied. */
      bFormatKnown = TRUE ;
      if ( msg.readItem( trigger ) ) {

        /* The trigger message has been extricated from the */
        /* buffer and transformed into a trigger object.    */
        /* Process the information in the trigger object. */
        ...
      }

      /* There is usually nothing after a trigger message. */
    }

    if ( msg.formatIs( FMT_USERCLASS ) ) {
      UserClass object ;
      /* The next item is an item of a user-defined class.     */
      /* For the next statement to work and return TRUE,       */
      /* the correct class of object pointer must be supplied. */
      bFormatKnown = TRUE ;

      if ( msg.readItem( object ) ) {
        /* The user-defined data has been extricated from the */
        /* buffer and transformed into a user-defined object. */

        /* Process the information in the user-defined object. */
        ...
      }

      /* Continue looking for further items. */
    }
    if ( ! bFormatKnown ) {
      /* There remains data that is not associated with a specific*/
      /* item class.                                               */
      char * pszDataPointer = msg.dataPointer( );       /* Address.*/
      int iDataLength = msg.dataLength( );              /* Length. */

      /* The encoding and character set for the remaining data are */
      /* reflected in the attributes of the message object, even   */
      /* if a dead-letter header was present.                      */
      ...

    }

  }
}
```

Bu örnekte FMT_USERCLASS , `UserClass`sınıfındaki bir nesneyle ilişkili 8 karakterlik biçim adını gösteren bir sabittir ve uygulama tarafından tanımlanır.

`UserClass` , ImqItem sınıfından türetilir (bkz. ImqItem C++ sınıfı) ve sanal **copyOut** ve **pasteIn** yöntemlerini o sınıftan uygular.

The next two examples show code from the ImqDeadLetterHeader class (see ImqDeadLetterHeader C++ sınıfı). İlk örnekte, özel kapsüllenmiş ileti-*yazı* kodu gösterilir.

```
// Insert a dead-letter header.
// Return TRUE if successful.
ImqBoolean ImqDeadLetterHeader :: copyOut ( ImqMessage & msg ) {
  ImqBoolean bSuccess ;
  if ( msg.moreBytes( sizeof( omqdlh ) ) ) {
    ImqCache cacheData( msg ); // Preserve original message content.
    // Note original message attributes in the dead-letter header.
    setEncoding( msg.encoding( ) );
    setCharacterSet( msg.characterSet( ) );
    setFormat( msg.format( ) );

    // Set the message attributes to reflect the dead-letter header.
    msg.setEncoding( MQENC_NATIVE );
    msg.setCharacterSet( MQCCSI_Q_MGR );
    msg.setFormat( MQFMT_DEAD_LETTER_HEADER );
    // Replace the existing data with the dead-letter header.
    msg.clearMessage( );
    if ( msg.write( sizeof( omqdlh ), (char *) & omqdlh ) ) {
      // Append the original message data.
      bSuccess = msg.write( cacheData.messageLength( ),
                            cacheData.bufferPointer( ) );
    } else {
      bSuccess = FALSE ;
    }
  } else {
    bSuccess = FALSE ;
  }
  // Reflect and cache error in this object.
  if ( ! bSuccess ) {
    setReasonCode( msg.reasonCode( ) );
    setCompletionCode( msg.completionCode( ) );
  }

  return bSuccess ;
}
```

İkinci örnekte, özel olarak kapsüllenmiş ileti-*okuma* kodu gösterilir.

```
// Read a dead-letter header.
// Return TRUE if successful.
ImqBoolean ImqDeadLetterHeader :: pasteIn ( ImqMessage & msg ) {
  ImqBoolean bSuccess = FALSE ;

  // First check that the eye-catcher is correct.
  // This is also our guarantee that the "character set" is correct.
  if ( ImqItem::structureIdIs( MQDLH_STRUC_ID, msg ) ) {
    // Next check that the "encoding" is correct, as the MQDLH
    // contains numeric data.
    if ( msg.encoding( ) == MQENC_NATIVE ) {

      // Finally check that the "format" is correct.
      if ( msg.formatIs( MQFMT_DEAD_LETTER_HEADER ) ) {
        char * pszBuffer = (char *) & omqdlh ;
        // Transfer the MQDLH from the message and move pointer on.
        if ( bSuccess = msg.read( sizeof( omdlh ), pszBuffer ) ) {
          // Update the encoding, character set and format of the
          // message to reflect the remaining data.
          msg.setEncoding( encoding( ) );
          msg.setCharacterSet( characterSet( ) );
          msg.setFormat( format( ) );
        } else {

          // Reflect the cache error in this object.
          setReasonCode( msg.reasonCode( ) );
          setCompletionCode( msg.completionCode( ) );
        }
      } else {
        setReasonCode( MQRC_INCONSISTENT_FORMAT );
        setCompletionCode( MQCC_FAILED );
      }
    } else {
      setReasonCode( MQRC_ENCODING_ERROR );
      setCompletionCode( MQCC_FAILED );
```

```
    {
  } else {
    setReasonCode( MQRC_STRUC_ID_ERROR );
    setCompletionCode( MQCC_FAILED );
  }

  return bSuccess ;
}
```

Otomatik arabellekle arabellek depolama alanı *uçucu*olur. Yani, arabellek verileri her **get** yöntemi çağrısından sonra farklı bir fiziksel yerde tutulabilir. Bu nedenle, her zaman arabellek verilerine başvurulsa, ileti verilerine erişmek için **bufferPointer** ya da **dataPointer** yöntemlerini kullanın.

Bir programın ileti verilerini almak için sabit bir alanı bir kenara koymasını isteyebilirsiniz. Bu durumda, **get** yöntemini kullanmadan önce **useEmptyBuffer** yöntemini çağırın.

Sabit olmayan, otomatik olmayan alan, iletileri büyüklük üst sınırına sınırlar. Bu nedenle, ImqGetMessageOptions nesnesinin MQGMO_ACCEPT_TRUNCATED_MSG seçeneğini göz önünde bulundurmanız önemlidir. Bu seçenek belirlenmezse (varsayılan değer), MQRC_TRUNCATED_MSG_FAILED neden kodu beklenebilir. Bu seçenek belirlenirse, uygulamanın tasarımına bağlı olarak MQRC_TRUNCATED_MSG_KABUL edilir neden kodu beklenebilir.

Sonraki örnekte, iletilerin nasıl alınabileceği, sabit bir depolama alanının nasıl kullanılabileceği gösterilmektedir:

```
char * pszBuffer = new char[ 100 ];

msg.useEmptyBuffer( pszBuffer, 100 );
gmo.setOptions( MQGMO_ACCEPT_TRUNCATED_MSG );
queue.get( msg, gmo );

delete [ ] pszBuffer ;
```

In this code fragment, the buffer can always be addressed directly, with *pszBuffer*, as opposed to using the **bufferPointer** method. Ancak, genel amaçlı erişim için **dataPointer** yönteminin kullanılması daha iyi olur. Uygulama (ImqCache sınıf nesnesi değil), kullanıcı tanımlı (otomatik olmayan) bir arabelleği atmalıdır.

Dikkat: Boş değerli bir gösterge ve **useEmptyArabellek** ile sıfır uzunluklu, sıfır uzunluklu değişmez uzunluktaki bir arabelleğinin beklenebileceği gibi bir değer göstermiyor. Bu birleşim, önceki kullanıcı tanımlı arabelleği yoksayma isteği olarak yorumlanır ve bunun yerine otomatik arabellek kullanımına döndür.

C++ dilinde ölü-mektup kuyruğuna bir ileti yazılıyor

Ölü-mektup kuyruğuna bir ileti yazmak için kullanılan örnek program kodu.

Çok bölümli bir iletinin tipik bir durumu, bir ölü-harf üstbilgisi içerir. İşlenemeyen bir iletiden gelen veriler, dead-letter üstbilgisine eklenir.

```
ImqQueueManager mgr ;        // The queue manager.
ImqQueue queueIn ;           // Incoming message queue.
ImqQueue queueDead ;         // Dead-letter message queue.
ImqMessage msg ;             // Incoming and outgoing message.
ImqDeadLetterHeader header ; // Dead-letter header information.

// Retrieve the message to be rerouted.
queueIn.setConnectionReference( mgr );
queueIn.setName( MY_QUEUE );
queueIn.get( msg );

// Set up the dead-letter header information.
header.setDestinationQueueManagerName( mgr.name( ) );
header.setDestinationQueueName( queueIn.name( ) );
header.setPutApplicationName( /* ? */ );
header.setPutApplicationType( /* ? */ );
header.setPutDate( /* TODAY */ );
```

```
header.setPutTime( /* NOW */ );
header.setDeadLetterReasonCode( FB_APPL_ERROR_1234 );

// Insert the dead-letter header information. This will vary
// the encoding, character set and format of the message.
// Message data is moved along, past the header.
msg.writeItem( header );

// Send the message to the dead-letter queue.
queueDead.setConnectionReference( mgr );
queueDead.setName( mgr.deadLetterQueueName( ) );
queueDead.put( msg );
```

C++ dilinde IMS köprüsine ileti yazma

IMS köprüsine bir ileti yazmak için kullanılan örnek program kodu.

WebSphere MQ-IMS köprüsine gönderilen iletiler özel bir üstbilgi kullanabilir. IMS köprüsü üstbilgisine önek olarak olağan ileti verileri eklenir.

```
ImqQueueManager    mgr;         // The queue manager.
ImqQueue           queueBridge; // IMS bridge message queue.
ImqMessage         msg;         // Outgoing message.
ImqIMSBridgeHeader header;      // IMS bridge header.

// Set up the message.
//
// Here we are constructing a message with format
// MQFMT_IMS_VAR_STRING, and appropriate data.
//
msg.write( 2,       /* ? */ );         // Total message length.
msg.write( 2,       /* ? */ );         // IMS flags.
msg.write( 7,       /* ? */ );         // Transaction code.
msg.write( /* ? */, /* ? */ );         // String data.
msg.setFormat( MQFMT_IMS_VAR_STRING ); // The format attribute.

// Set up the IMS bridge header information.
//
// The reply-to-format is often specified.
// Other attributes can be specified, but all have default values.
//
header.setReplyToFormat( /* ? */ );

// Insert the IMS bridge header into the message.
//
// This will:
// 1) Insert the header into the message buffer, before the existing
//    data.
// 2) Copy attributes out of the message descriptor into the header,
//    for example the IMS bridge header format attribute will now
//    be set to MQFMT_IMS_VAR_STRING.
// 3) Set up the message attributes to describe the header, in
//    particular setting the message format to MQFMT_IMS.
//
msg.writeItem( header );

// Send the message to the IMS bridge queue.
//
queueBridge.setConnectionReference( mgr );
queueBridge.setName( /* ? */ );
queueBridge.put( msg );
```

C++ dilinde CICS köprüsünün bir ileti yazılması

CICS köprüsünün bir iletiyi yazmak için kullanılan örnek program kodu.

CICS köprüsünü kullanan z/OS için WebSphere MQ ' ya gönderilen iletiler özel bir üstbilgi gerektirir. CICS köprüsü üstbilgisine önek olarak olağan ileti verileri eklenir.

```
ImqQueueManager mgr ;         // The queue manager.
ImqQueue queueIn ;            // Incoming message queue.
ImqQueue queueBridge ;        // CICS bridge message queue.
```

```
ImqMessage msg ;              // Incoming and outgoing message.
ImqCicsBridgeHeader header ;  // CICS bridge header information.

// Retrieve the message to be forwarded.
queueIn.setConnectionReference( mgr );
queueIn.setName( MY_QUEUE );
queueIn.get( msg );

// Set up the CICS bridge header information.
// The reply-to format is often specified.
// Other attributes can be specified, but all have default values.
header.setReplyToFormat( /* ? */ );

// Insert the CICS bridge header information.  This will vary
// the encoding, character set and format of the message.
// Message data is moved along, past the header.
msg.writeItem( header );

// Send the message to the CICS bridge queue.
queueBridge.setConnectionReference( mgr );
queueBridge.setName( /* ? */ );
queueBridge.put( msg );
```

C++ dilinde iş üstbilgisiyle ileti yazma

z/OS Workload Manager tarafından yönetilen bir kuyruğa ilişkin ileti yazılmasına ilişkin örnek program kodu.

Messages sent to WebSphere MQ for z/OS, which are destined for a queue managed by the z/OS Workload Manager, require a special header. İş üstbilgisinin başında düzenli ileti verisi var.

```
ImqQueueManager mgr ;             // The queue manager.
ImqQueue queueIn ;                // Incoming message queue.
ImqQueue queueWLM ;               // WLM managed queue.
ImqMessage msg ;                  // Incoming and outgoing message.
ImqWorkHeader header ;            // Work header information

// Retrieve the message to be forwarded.
queueIn.setConnectionReference( mgr );
queueIn.setName( MY_QUEUE );
queueIn.get( msg );

// Insert the Work header information.  This will vary
// the encoding, character set and format of the message.
// Message data is moved along, past the header.
msg.writeItem( header );

// Send the message to the WLM managed queue.
queueWLM.setConnectionReference( mgr );
queueWLM.setName( /* ? */ );
queueWLM.put( msg );
```

Building WebSphere MQ C++ programs

Desteklenen derleyicilerin URL 'si, C++ programlarını ve örneklerini WebSphere MQ altyapılarında derlemek, bağlamak ve çalıştırmak için kullanılacak komutlarla birlikte listelenir.

The compilers for each supported platform and version of WebSphere MQ are listed at the WebSphere MQ system requirements page at IBM WebSphere MQ.

WebSphere MQ C++ programınızı derlemek ve bağlamak için gereken komut, kuruluş ve gereksinmelerinize bağlıdır. Aşağıdaki örneklerde, varsayılan WebSphere MQ kuruluşunu kullanan derleyicilerin bazıları için tipik derleme ve bağlantı komutları gösterilir.

Building C++ programs on AIX

XL C Enterprise Edition derleyicisini kullanarak AIX üzerinde WebSphere MQ C++ programları oluşturun.

İstemci

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

32 bitlik iş parçacıklı uygulama

```
xlC -o imqsputc_32 imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -limqc23ia -limqb23ia -lmqic
```

32 bitlik iş parçacıklı uygulama

```
xlC_r -o imqsputc_32_r imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -limqc23ia_r -limqb23ia_r -lmqic_r
```

64 bitlik iş parçacıklı uygulama

```
xlC -q64 -o imqsputc_64 imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -limqc23ia -limqb23ia -lmqic
```

64 bitlik iş parçacıklı uygulama

```
xlC_r -q64 -o imqsputc_64_r imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -limqc23ia_r -limqb23ia_r -lmqic_r
```

Sunucu

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

32 bitlik iş parçacıklı uygulama

```
xlC -o imqsput_32 imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -limqs23ia -limqb23ia -lmqm
```

32 bitlik iş parçacıklı uygulama

```
xlC_r -o imqsput_32_r imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -limqs23ia_r -limqb23ia_r -lmqm_r
```

64 bitlik iş parçacıklı uygulama

```
xlC -q64 -o imqsput_64 imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -limqs23ia -limqb23ia -lmqm
```

64 bitlik iş parçacıklı uygulama

```
xlC_r -q64 -o imqsput_64_r imqsput.cpp -qchars=signed -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -limqs23ia_r -limqb23ia_r -lmqm_r
```

C++ programlarını HP-UXüzerinde oluşturma

aC+ + ya da aCC derleyicisini kullanarak HP-UX üzerinde WebSphere MQ C++ programları oluşturun.

HP-UX Itaniumüzerinde, WebSphere MQ yalnızca Standart çalıştırma zamanını destekler. aCC derleyicisini kullanın.

- libimqi23bh.sl , Standart çalıştırma zamanı için WebSphere MQ C++ sınıflarını sağlar.
- Daha önceki yayınlarla uyumluluk için, libimqi23ah.sl öğesinden libimqi23bh.sl' ye simgesel bir bağlantı sağlanır.

IA64 (IPF)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci: IA64 (IPF)

32 bitlik iş parçacıklı uygulama

```
aCC -Wl,+b,: +e -D_HPUX_SOURCE -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -limqi23bh -lmqic
```

32 bitlik iş parçacıklı uygulama

```
aCC -Wl,+b,: +e -D_HPUX_SOURCE -o imqsputc_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -limqi23bh_r -lmqic_r -lpthread
```

64 bitlik iş parçacıklı uygulama

```
aCC +DD64 +e -D_HPUX_SOURCE -o imqsputc_64 imqsput.cpp
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -limqi23bh -lmqic
```

64 bitlik iş parçacıklı uygulama

```
aCC +DD64 +e -D_HPUX_SOURCE -o imqsputc_64_r imqsput.cpp
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -limqi23bh_r
-lmqic_r
-lpthread
```

Sunucu: IA64 (IPF)

32 bitlik iş parçacıklı uygulama

```
aCC -Wl,+b,: +e -D_HPUX_SOURCE -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -limqi23bh -lmqm
```

32 bitlik iş parçacıklı uygulama

```
aCC -Wl,+b,: +e -D_HPUX_SOURCE -o imqsput_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -L/usr/lib/hpux32 -limqi23bh_r -lmqm_r -lpthread
```

64 bitlik iş parçacıklı uygulama

```
aCC +DD64 +e -D_HPUX_SOURCE -o imqsput_64 imqsput.cpp
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -limqi23bh -lmqm
```

64 bitlik iş parçacıklı uygulama

```
aCC +DD64 +e -D_HPUX_SOURCE -o imqsput_64_r imqsput.cpp
-IMQ_INSTALLATION_PATH/inc -LMQ_INSTALLATION_PATH/lib64 -L/usr/lib/hpux64 -limqi23bh_r
-lmqm_r
-lpthread
```

Building C++ programs on Linux

GNU g + + derleyicisini kullanarak Linux üzerinde WebSphere MQ C++ programları oluşturun.

System p

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci: System p

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -o imqsputc_32 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl
-limqb23gl -lmqic
```

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -o imqsputc_r32 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl_r
-limqb23gl_r -lmqic_r
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -o imqsputc_64 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqc23gl -limqb23gl -lmqic
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -o imqsputc_r64 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqc23gl_r -limqb23gl_r -lmqic_r
```

Sunucu: System p

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -o imqsput_32 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl
-limqb23gl -lmqm
```

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -o imqsput_r32 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl_r
-limqb23gl_r -lmqm_r
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -o imqsput_64 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqs23gl -limqb23gl -lmqm
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -o imqsput_r64 imqsput.cpp -fsigned-char -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqs23gl_r -limqb23gl_r -lmqm_r
```

System z

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci: System z

32 bitlik iş parçacıklı uygulama

```
g++ -m31 -fsigned-char -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl -limqb23gl -lmqic
```

32 bitlik iş parçacıklı uygulama

```
g++ -m31 -fsigned-char -o imqsputc_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqc23gl_r -limqb23gl_r -lmqic_r
-lpthread
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqc23gl -limqb23gl -lmqic
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsputc_64_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64  -Wl,-rpath=/usr/lib64
-limqc23gl_r -limqb23gl_r -lmqic_r -lpthread
```

Sunucu: System z

32 bitlik iş parçacıklı uygulama

```
g++ -m31 -fsigned-char -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib  -Wl,-rpath=/usr/lib
-limqs23gl -limqb23gl -lmqm
```

32 bitlik iş parçacıklı uygulama

```
g++ -m31 -fsigned-char -o imqsput_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib
-limqs23gl_r -limqb23gl_r -lmqm_r -lpthread
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqs23gl -limqb23gl -lmqm
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsput_64_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64
-limqs23gl_r -limqb23gl_r -lmqm_r -lpthread
```

System x (32 bit)

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci: System x (32 bit)

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -fsigned-char -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -LMQ_INSTALLATION_PATH/lib
-Wl,
-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -limqc23gl -limqb23gl -lmqic
```

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -fsigned-char -o imqsputc_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -limqc23gl_r -limqb23gl_r
-lmqic_r -lpthread
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -LMQ_INSTALLATION_PATH/
lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -limqc23gl -limqb23gl
-lmqic
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsputc_64_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -LMQ_INSTALLATION_PATH/
lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -limqc23gl_r -limqb23gl_r
-lmqic_r -lpthread
```

Sunucu: System x (32 bit)

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -fsigned-char -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -limqs23gl -limqb23gl -lmqm
```

32 bitlik iş parçacıklı uygulama

```
g++ -m32 -fsigned-char -o imqsput_32_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATHlib -Wl,-rpath=MQ_INSTALLATION_PATH/lib -LMQ_INSTALLATION_PATH/lib
-Wl,-rpath=MQ_INSTALLATION_PATH/lib -Wl,-rpath=/usr/lib -limqs23gl_r -limqb23gl_r
-lmqm_r -lpthread
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -LMQ_INSTALLATION_PATH/
lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -limqs23gl -limqb23gl -lmqm
```

64 bitlik iş parçacıklı uygulama

```
g++ -m64 -fsigned-char -o imqsput_64_r imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -LMQ_INSTALLATION_PATH/
lib64
-Wl,-rpath=MQ_INSTALLATION_PATH/lib64 -Wl,-rpath=/usr/lib64 -limqs23gl_r -limqb23gl_r
-lmqm_r -lpthread
```

Solaris üzerinde C++ programları oluşturulması

Sun ONE derleyicisini kullanarak Solaris üzerinde WebSphere MQ C++ programları oluşturun.

SPARC

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Müşteri: SPARC

32 bit uygulama

```
CC -xarch=v8plus -mt -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
```

```
-LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqc23as -limqb23as
-lmqic -lsocket -lnsl -ldl
```

64 bit uygulama

```
CC -xarch=v9 -mt -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqc23as
-limqb23as
-lmqic -lsocket -lnsl -ldl
```

Sunucu: SPARC

32 bit uygulama

```
CC -xarch=v8plus -mt -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqs23as -limqb23as
-lmqm -lsocket -lnsl -ldl
```

64 bit uygulama

```
CC -xarch=v9 -mt -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqs23as
-limqb23as
-lmqm -lsocket -lnsl -ldl
```

x86-64

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci: x86-64

32 bit uygulama

```
CC -xarch=386 -mt -o imqsputc_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqc23as -limqb23as
-lmqic -lsocket -lnsl -ldl
```

64 bit uygulama

```
CC -xarch=amd64 -mt -o imqsputc_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqc23as
-limqb23as
-lmqic -lsocket -lnsl -ldl
```

Sunucu: x86-64

32 bit uygulama

```
CC -xarch=386 -mt -o imqsput_32 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib -RMQ_INSTALLATION_PATH/lib -R/usr/lib/32 -limqs23as -limqb23as
-lmqm -lsocket -lnsl -ldl
```

64 bit uygulama

```
CC -xarch=amd64 -mt -o imqsput_64 imqsput.cpp -IMQ_INSTALLATION_PATH/inc
-LMQ_INSTALLATION_PATH/lib64 -RMQ_INSTALLATION_PATH/lib64 -R/usr/lib/64 -limqs23as
-limqb23as
-lmqm -lsocket -lnsl -ldl
```

C++ programlarını Windowsüzerinde oluşturma

Microsoft Visual Studio C++ derleyicisini kullanarak Windows üzerinde WebSphere MQ C++ programları oluşturun.

32 bit uygulamalarla kullanım için kitaplık (.lib) dosyaları ve dll dosyaları *MQ_INSTALLATION_PATH*/Tools/Libiçinde kurulur, 64 bit uygulamalarla kullanmak için dosyalar *MQ_INSTALLATION_PATH*/Tools/Lib64içinde kurulur. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

İstemci

```
cl -MD imqsput.cpp /Feimqsputc.exe imqb23vn.lib imqc23vn.lib
```

Sunucu

```
cl -MD imqsput.cpp /Feimqsput.exe imqb23vn.lib imqs23vn.lib
```

Java için WebSphere MQ sınıflarının kullanılması

Java için WebSphere MQ sınıfları, Java ortamında WebSphere MQ ' yı kullanmanıza olanak sağlar. A Java application can use either WebSphere MQ classes for Java or WebSphere MQ classes for JMS to access WebSphere MQ resources.

Java için WebSphere MQ class for Java uygulaması aşağıdaki gibi bir Java uygulamasına izin verir:

- Bir WebSphere MQ istemcisi olarak WebSphere MQ ' a bağlanma
- Doğrudan bir WebSphere MQ kuyruk yöneticisine bağlan

Java için WebSphere MQ sınıfları, yerel WebSphere MQ API ' yı (Message Queue Interface; İleti Kuyruğu Arabirimi) (MQI) sarmala.

Java için WebSphere MQ sınıfları, C++ ve .NET arabirimlerine benzer bir nesne modelini WebSphere MQ' ya kullanır.

Java için WebSphere MQ sınıflarını neden kullanmalıyım?

Kuruluşunuzda aşağıdaki noktalar önem gösteriyorsa, Java için WebSphere MQ sınıflarını kullanmayı düşünün:

- Java için WebSphere MQ sınıfları, yerel WebSphere MQ API ' yı (Message Queue Interface; İleti Kuyruğu Arabirimi) (MQI) sarmala.
  - Yordamsal dillerdeki MQI ' nin kullanılmasına alışıksanız, bu bilgiyi Java ortamına aktarabilirsiniz.
  - You can exploit the full range of features of WebSphere MQ, beyond those available through JMS.
- Java için WebSphere MQ sınıfları, C++ ve .NET arabirimlerine benzer bir nesne modelini WebSphere MQ' ya kullanır. Bu arabirimleri alıyorsanız, bu bilgiyi Java ortamına aktarabilirsiniz.

Not: Otomatik istemci yeniden bağlantısı, Java için WebSphere MQ sınıfları tarafından desteklenmez.

Java için WebSphere MQ sınıflarıyla çalışmaya başlama

Bu konu derlemi, Java ve kullanımları için WebSphere MQ sınıflarına genel bir bakış sağlar.

Java için WebSphere MQ sınıfları nedir?

Java içinWebSphere MQ sınıfları, Java ortamında WebSphere MQ olanağını kullanmanıza olanak sağlar.

Java içinWebSphere MQ class for Java uygulaması aşağıdaki gibi bir Java uygulamasına izin verir:

- Bir WebSphere MQ istemcisi olarak WebSphere MQ ' a bağlanma
- Doğrudan bir WebSphere MQ kuyruk yöneticisine bağlan

Java içinWebSphere MQ sınıfları, yerel WebSphere MQ API ' yı (Message Queue Interface; İleti Kuyruğu Arabirimi) (MQI) sarmala.

Java içinWebSphere MQ sınıfları, C++ ve .NET arabirimlerine benzer bir nesne modelini WebSphere MQ' ya kullanır.

Java için WebSphere MQ sınıflarını neden kullanmalıyım?

A Java application can use either WebSphere MQ classes for Java or WebSphere MQ classes for JMS to access WebSphere MQ resources. Java için WebSphere MQ sınıflarının kullanılmasıyla ilgili bir dizi avantaj vardır.

Kuruluşunuzda aşağıdaki noktalar önem gösteriyorsa, Java için Websphere MQ sınıflarını kullanmayı düşünün:

- Java içinWebSphere MQ sınıfları, yerel WebSphere MQ API ' yı (Message Queue Interface; İleti Kuyruğu Arabirimi) (MQI) sarmala.
  - Yordamsal dillerdeki MQI ' nin kullanılmasına alışıksanız, bu bilgiyi Java ortamına aktarabilirsiniz.
  - You can exploit the full range of features of WebSphere MQ, beyond those available through JMS.
- Java içinWebSphere MQ sınıfları, C++ ve .NET arabirimlerine benzer bir nesne modelini WebSphere MQ' ya kullanır. Bu arabirimleri alıyorsanız, bu bilgiyi Java ortamına aktarabilirsiniz.

Java için WebSphere MQ sınıflarına ilişkin bağlantı seçenekleri

Java için WebSphere MQ sınıfları istemci ya da bağ tanımları kipinde bağlanabilirler.

Programlanabilir seçenekler, Java için WebSphere MQ sınıflarının WebSphere MQ ' ya aşağıdaki yöntemlerden birini kullanarak bağlanmasını sağlar:

- Bir WebSphere MQ MQI istemcisi olarak, İletim Denetimi İletişim Kurası/Internet Protocol (TCP/IP)
- Bağ tanımları kipinde, Java Native Interface (JNI) kullanarak doğrudan WebSphere MQ ' ya bağlanılması

İstemciler z/OSüzerinde çalıştırılamaz, ancak diğer platformlardaki istemciler, Client Attach olanağı kuruluysa, z/OS kuyruk yöneticisi için bir WebSphere MQ ' ya bağlanabilir.

Aşağıdaki kısımlarda, istemci kipi ve bağ tanımları kipi bağlantı seçenekleri daha ayrıntılı olarak açıklanmıştır.

İstemci bağlantısı

İstemci kipinde bir kuyruk yöneticisine bağlanmak için, Java uygulamasına ilişkin bir WebSphere MQ sınıfları, kuyruk yöneticisinin çalıştığı sistemde ya da farklı bir sistemde çalıştırılabilir. Her durumda, Java için WebSphere MQ sınıfları, TCP/IP üzerinden kuyruk yöneticisine bağlanır.

Java uygulamasına ilişkin bir WebSphere MQ sınıfları, istemci kipini kullanarak desteklenen bir kuyruk yöneticisine bağlanabilir.

İstemci kipi bağlantılarını kullanmak üzere uygulamaların nasıl yazılabilmesiyle ilgili daha fazla bilgi için bkz. “Java bağlantı kipleri içinWebSphere MQ sınıfları” sayfa 641.

Bağ tanımları bağlantısı

Bağ tanımları kipinde kullanıldığında, Java için WebSphere MQ sınıfları Java Native Interface 'i (JNI), bir ağ üzerinden iletişim kurmak yerine, doğrudan var olan kuyruk yöneticisi API ' sına çağrılacak şekilde kullanır. Çoğu ortamda, bağ tanımları kipindeki bağlantı, TCP/IP iletişiminin maliyetini önleterek, istemci kipinde bağlanmaktan daha iyi WebSphere MQ sınıfları için daha iyi başarım sağlar.

Bağ tanımları kipinde bağlanmak için Java için WebSphere MQ sınıflarını kullanan uygulamaların, bağlantı kurdukları kuyruk yöneticiyle aynı sistemde çalıştırılması gerekir.

Java uygulaması için WebSphere MQ sınıflarını çalıştırmak için kullanılan Java Runtime Environment, Java kitaplıkları için WebSphere MQ sınıflarını yüklemek üzere yapılandırılmalıdır; ek bilgi için Java kitaplıklarına ilişkin WebSphere MQ sınıflarına bakın.

Bağ tanımları kipi bağlantılarını kullanmak üzere uygulamaların nasıl yazılabilmesiyle ilgili daha fazla bilgi için bkz. “Java bağlantı kipleri içinWebSphere MQ sınıfları” sayfa 641.

Java için WebSphere MQ sınıflarına ilişkin önkoşullar

Java için WebSphere MQ sınıflarını kullanmak için bazı diğer yazılım ürünlerine gereksiniminiz vardır.

Java için WebSphere MQ sınıflarına ilişkin önkoşullarla ilgili en son bilgiler için WebSphere MQ README (Beni Oku) dosyasına bakın.

Java uygulamaları için WebSphere MQ sınıfları geliştirmek için bir Java Geliştirme Takımı 'na (JDK) gerek vardır. İşletim sisteminizle desteklenen JDKS ' lerin ayrıntıları, IBM WebSphere MQadresindeki WebSphere MQ sistem gereksinimleri sayfasında bulunabilir.

Java uygulamaları için WebSphere MQ sınıflarını çalıştırmak için aşağıdaki yazılım bileşenlerine gereksinim duyarsınız:

- Bir kuyruk yöneticisine bağlanan uygulamalar için bir WebSphere MQ kuyruk yöneticisi
- Uygulamaları çalıştırdığınız her sistem için bir Java Runtime Environment (JRE). A suitable JRE is supplied with WebSphere MQ.

FIPS 140-2 sertifikalı şifreleme modülleri kullanmak için SSL bağlantısına gereksinim duyuyorsanız, IBM Java JSSE FIPS sağlayıcısına (IBMJSSEFIPS) gerek duyarsınız. Every IBM JDK and JRE at Version 1.4.2 or later contains IBMJSSEFIPS.

You can use Internet Protocol Version 6 (IPv6) addresses in your WebSphere MQ classes for Java applications IPv6 ise supported by your Java virtual machine (JVM) and the TCP/IP implementation on your operating system.

Java için WebSphere MQ sınıflarının kurulması ve yapılandırılması

Bu kısımda, Java için WebSphere MQ sınıfları kurulurken yaratılan dizinler ve dosyalar anlatılır ve kuruluştan sonra Java için WebSphere MQ sınıflarının nasıl yapılandırılacağı anlatılır.

Java için WebSphere MQ sınıfları için kurulu olan öğe

Java için WebSphere MQ sınıflarının en son sürümü WebSphere MQile birlikte kurulur. Bunun yapıldığından emin olmak için varsayılan kuruluş seçeneklerini geçersiz kılmanız gerekebilir.

WebSphere MQ ürününün kurulmasıyla ilgili ek bilgi edinmek için aşağıdaki başlara bakın:

WebSphere MQ sunucusunun kurulması
IBM WebSphere MQ istemcisi kurulması

Java için WebSphere MQ sınıfları Java arşivi (JAR) dosyaları (com.ibm.mq.jarve com.ibm.mq.jmqi.jar) içerisindir.

Programlanabilir Komut Biçimi (PCF) gibi standart ileti üstbilgileri için destek, com.ibm.mq.headers.jarJAR dosyasında bulunur.

Programlanabilir Komut Biçimi (PCF) desteği, com.ibm.mq.pcf.jaradlı JAR dosyasında bulunur.

Java JAR dosyaları için WebSphere MQ sınıflarının kurulması ve büyütülmesi

The only supported way to get the WebSphere MQ classes for Java JAR files onto a system, is to install either the WebSphere MQ product, or the WebSphere MQ MQI client SupportPac, or to use a software management tool such as Apache Maven, for more information see “IBM WebSphere MQ classes for Java ve yazılım yönetimi araçları” sayfa 637.

Bir yazılım yönetimi aracı kullanmıyorsanız, WebSphere MQ sınıflarını Java JAR dosyaları için diğer makinelerden taşımayın ya da kopyalamayın.

- Düzeltme paketleri, JAR dosyalarının başka bir makineden kopyalandığı bir "kuruluşa" uygulanamaz ve tüm JAR dosyalarının birbirleri ile aynı adımda tutulmasını ve uyumlu düzeylerde olduğundan emin olmak daha zor duruma getirir.
- Makineler arasında JMS JAR dosyalarına ilişkin WebSphere MQ sınıflarının kopyalanması aynı makinede bulunan dosyaların birden çok kopyasının da sonuçlanmasına neden olabilir; bu da kodun bakılmasına ve sorunlara hata ayıklanmasına neden olabilir.

Uygulama arşivleri içinde Java JAR dosyaları için WebSphere MQ sınıflarını eklemeyin.

- Bir WebSphere MQ düzeltme paketi kullanılarak, Java için WebSphere MQ sınıflarına ilişkin güncellemeler uygulanamaz.
- It is not be possible for IBM Support to easily determine the version of the WebSphere MQ classes for Java that are being used by the application.
- Aynı Java Runtime Environment içinde çalışan birden çok uygulamanın Java için WebSphere MQ sınıflarının farklı sürümleri varsa, Java için WebSphere MQ sınıflarının birden çok sürümü Java Runtime Environment 'a aynı anda yüklenirse, sorunlar ortaya çıkabilir.
- Bir uygulama, bir kuyruk yöneticisine bağlanmak için BAĞKUR iletisini kullanıyorsa, kuyruk yöneticisine yapılan tüm ana yükseltmeler de, uygulamanın Java için WebSphere MQ sınıflarının karşılık gelen düzeyini içerecek şekilde güncellenmesini gerektirir.

  Örneğin, bir kuyruk yöneticisi WebSphere MQ Sürüm 7.1 düzeyine büyütülürse, bu durumda kuyruk yöneticisine bağlanan herhangi bir uygulamanın, Java için WebSphere MQ Sürüm 7.1 sınıflarını içerecek şekilde güncellenmesi de gerekir.

Java için WebSphere MQ sınıfları ile aşağıdaki Java kitaplığı dağıtılır:

- connector.jar (Sürüm 1.0)

Postcard adlı örnek uygulama com.ibm.mq.postcard.jaradlı JAR dosyasında bulunuyor.

Javadoc aracı, JMS API ' leri için Java ve WebSphere MQ sınıflarına ilişkin WebSphere MQ sınıflarının belirtimlerini içeren HTML sayfalarını oluşturmak için kullanılır. HTML sayfaları, JMS kuruluş dizini için WebSphere MQ sınıflarının doc altdizininde yer alıyor. UNIX, Linuxve Windows sistemlerinde doc alt dizini, tek tek HTML sayfalarını içeriradlı bir dosyada yer alır.

Kuruluş tamamlanınca, dosyalar ve örnekler “Java için WebSphere MQ sınıflarına ilişkin kuruluş dizinleri” sayfa 630içinde gösterilen konumlara kurulur.

Kuruluştan sonra, Windowsdışındaki herhangi bir altyapıda, ortam değişkenlerinizi “Java için WebSphere MQ sınıflarıyla ilgili ortam değişkenleri” sayfa 631içinde açıklandığı gibi güncellemeniz gerekir.

Java için WebSphere MQ sınıflarına ilişkin kuruluş dizinleri

Java dosyaları için WebSphere MQ sınıfları, altyapıya göre farklı konumlara kurulur.

Çizelge 82 sayfa 630 , Java dosyaları için WebSphere MQ sınıflarının kurulu olduğu yeri gösterir.

Çizelge 82. Java kuruluş dizinlerine ilişkin WebSphere MQ sınıfları

Altyapı	**Dizin**
AIX	*MQ_INSTALLATION_PATH*/java/lib
HP-UX, Linuxve Solaris	*MQ_INSTALLATION_PATH*/java/lib
Pencereler	*MQ_INSTALLATION_PATH*\java\lib
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

Installation Verification Programs (IVP) gibi bazı örnek uygulamalar WebSphere MQile birlikte sağlanır. Çizelge 83 sayfa 631 Örnek uygulamaların nerede kurulu olduğunu gösterir. Java örnekleri için

WebSphere MQ sınıfları, `wmqjava`adlı bir altdizinde yer alıyor. PCF örnekleri, `pcf`adlı bir altdizinde yer alıyor.

Çizelge 83. Örnek dizinleri	
Altyapı	**Dizin**
AIX	*MQ_INSTALLATION_PATH*/samp/wmqjava/
HP-UX, Linuxve Solaris	*MQ_INSTALLATION_PATH*/samp/wmqjava/
Pencereler	*MQ_INSTALLATION_PATH*\tools\wmqjava\
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

Java için WebSphere MQ sınıflarıyla ilgili ortam değişkenleri

Java uygulamaları için WebSphere MQ sınıflarını çalıştırmak istiyorsanız, sınıf yollarının Java ve örnek dizinlerine ilişkin WebSphere MQ sınıflarını içermesi gerekir.

Java uygulamalarının çalışması için WebSphere MQ sınıfları için, bunların sınıf yolu Java dizini için uygun WebSphere MQ sınıflarını içermelidir. Örnek uygulamaları çalıştırmak için, sınıf yolunun uygun örnek dizinlerini de içermesi gerekir. Bu bilgi Java çağırma komutunda ya da CLASSPATH ortam değişkeninde sağlanabilir.

Çizelge 84 sayfa 631 , örnek uygulamalar da içinde olmak üzere, Java uygulamalarına ilişkin WebSphere MQ sınıflarını çalıştırmak için her altyapıda kullanılacak uygun CLASSPATH ayarını gösterir.

Çizelge 84. Java örnek uygulamaları için WebSphere MQ sınıfları da içinde olmak üzere, Java uygulamaları için WebSphere MQ sınıflarını çalıştırmak için CLASSPATH ayarı	
Altyapı	**SPATH ayarı**
AIX	CLASSPATH=*MQ_INSTALLATION_PATH*/java/lib/com.ibm.mq.jar: *MQ_INSTALLATION_PATH*/samp/wmqjava/samples:
HP-UX, Linuxve Solaris	CLASSPATH=*MQ_INSTALLATION_PATH*/java/lib/com.ibm.mq.jar: *MQ_INSTALLATION_PATH*/samp/wmqjava/samples:
Pencereler	CLASSPATH=*MQ_INSTALLATION_PATH*\Java\lib\com.ibm.mq.jar; *MQ_INSTALLATION_PATH*\tools\wmqjava\samples;
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

-Xlint seçeneğini kullanarak derleyip derlendiyseniz, com.ibm.mq.ese.jar dosyasının yok olduğunu bildiren bir ileti görebilirsiniz. Uyarıyı yoksayabilirsiniz. Bu dosya yalnızca IBM WebSphere MQ Advanced Message Securityürününü kurdıysanız bulunur.

Java için WebSphere MQ sınıflarıyla sağlanan komut dosyaları aşağıdaki ortam değişkenlerini kullanır:

MQ_JAVA_DATA_PATH
Bu ortam değişkeni, günlük ve izleme çıkışına ilişkin dizini belirtir.

MQ_JAVA_INSTALL_PATH
This environment variable specifies the directory where WebSphere MQ classes for Java are installed, as shown in Java kuruluş dizinlerine ilişkinWebSphere MQ sınıfları.

MQ_JAVA_LIB_PATH
This environment variable specifies the directory where the WebSphere MQ classes for Java libraries are stored, as shown in Her altyapıya ilişkin Java kitaplıklarına ilişkin WebSphere MQ sınıflarının yeri. Java için WebSphere MQ sınıflarıyla (IVTRun gibi) verilen bazı komut dosyaları bu ortam değişkenini kullanır.

Pencereler' ta, kuruluş sırasında tüm ortam değişkenleri otomatik olarak ayarlanır. Başka bir platform üzerinde, onları kendiniz ayarlamalısınız.Bir UNIX sisteminde, ortam değişkenlerini ayarlamak için **setjmsenv** komut dosyasını (32 bit JVM kullanıyorsanız) ya da **setjmsenv64** (64 bit JVM kullanıyorsanız) kullanabilirsiniz. AIX, HP-UX, Linuxve Solaris üzerinde bu komut dosyaları *MQ_INSTALLATION_PATH*/java/bin dizininde bulunur.

Java kitaplıklarına ilişkin IBM WebSphere MQ sınıfları

Java kitaplıklarına ilişkin IBM WebSphere MQ sınıflarının yeri altyapıya göre değişir. Bir uygulamayı başlattığınızda bu konumu belirtin.

To specify the location of the Java Native Interface (JNI) libraries, start your application using a **java** command with the following format:

```
java -Djava.library.path=library_path application_name
```

Burada *kitaplık_yolu* , JNI kitaplıklarının bulunduğu Java kitaplıklarına ilişkin WebSphere MQ sınıflarının yoludur. Çizelge 85 sayfa 632 her bir platform için Java kitaplıkları için WebSphere MQ sınıflarının yerini gösterir.

Çizelge 85. Her platform için Java kitaplıklarına ilişkin WebSphere MQ sınıflarının konumu

Altyapı	**Java kitaplıkları için WebSphere MQ sınıflarını içeren dizin**
AIX	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)
HP-UX Linux (POWER, x86-64 ve zSeries s390x platformları) Solaris (x86-64 ve SPARC platformları)	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)
Linux (x86 platformu)	*MQ_INSTALLATION_PATH*/java/lib
Pencereler	*MQ_INSTALLATION_PATH*\Java\lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*\Java\lib64 (64 bit kitaplık)
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

Not:

1. AIX, HP-UX, Linux (Power platform) ya da Solaris üzerinde, 32 bit kitaplıklarını ya da 64 bit kitaplıklarını kullanın. 64 bit kullanan kitaplıkları 64 bit altyapıda bir JVM (64 bit Java sanal makinesi) içinde çalıştırıyorsanız, 64 bit kitaplıklarını kullanın. Tersi durumda, 32 bit kitaplıklarını kullanın.
2. On Pencereler, you can use the PATH environment variable to specify the location of the WebSphere MQ classes for Java libraries instead of specifying their location on the **java** command.
3. IBM iüzerinde bağ tanımları kipinde Java için WebSphere MQ sınıflarını kullanmak için, QMQMJAVA kitaplığının kitaplık listenizde olduğundan emin olun.

İlgili görevler

Java için WebSphere MQ sınıflarının kullanılması

IBM WebSphere MQ classes for Javaüzerinde OSGi desteği

OSGi, uygulamaların kod paketleri olarak konuşlandırılmasını destekleyen bir çerçeve sağlar. IBM WebSphere MQ classes for Java ' nin bir parçası olarak bir OSGi kod paketi sağlanır.

OSGi, kod paketleri biçiminde gelen uygulamaların konuşlandırılmasını destekleyen genel amaçlı, güvenli ve yönetilenJava çerçevesini sağlar. OSGi uyumlu aygıtlar, paketleri yükleyebilir ve kurabilir ve artık gerekli olmadığında bunları kaldırabilir. Çerçeve, kod paketlerinin dinamik ve ölçeklenebilir bir şekilde kurulup güncellenmesini yönetir.

IBM WebSphere MQ classes for Java. Aşağıdaki OSGi paketini içerir.

com.ibm.mq.osgi.java_< sürüm numarası > .jar
Uygulamaların IBM WebSphere MQ classes for Javakullanmasını sağlamak için JAR dosyaları.

Burada < sürüm numarası >, kurulu olan WebSphere MQ ' un sürüm sayısıdır.

Paket, IBM WebSphere MQ kurulumunuzun `java/lib/OSGi` alt dizinine ya da Windows 'ta `java\lib\OSGi` klasörüne kurulur.

Diğer dokuz kod paketi de IBM WebSphere MQ kurulumunuzun `java/lib/OSGi` alt dizinine ya da Windows 'ta `java\lib\OSGi` klasörüne kurulur. Bu paketler IBM WebSphere MQ classes for JMS ' in bir parçasıdır ve IBM WebSphere MQ classes for Java kod paketi yüklü olan bir OSGi yürütme ortamına yüklenmemelidir. IBM WebSphere MQ classes for Java OSGi kod paketi, IBM WebSphere MQ classes for JMS kod paketleri de yüklü olan bir OSGi yürütme ortamına yüklenirse, aşağıdakiler gibi hatalar:

```
java.lang.ClassCastException: com.ibm.mq.MQException incompatible with com.ibm.mq.MQException
```

oluşan, IBM WebSphere MQ classes for Java kod paketi ya da IBM WebSphere MQ classes for JMS kod paketlerini kullanan uygulamalar çalıştırılır.

IBM WebSphere MQ classes for Java için OSGi paketi OSGi Yayın Düzeyi 4 belirtimine yazıldı; bir OSGi Yayın Düzeyi 3 ortamında çalışmıyor.

OSGi yürütme ortamı gereken DLL kütüklerini ya da paylaşılan kitaplıkları bulabilmesi için, sistem yolunu ya da kitaplık yolunu doğru olarak ayarlamalısınız.

IBM WebSphere MQ classes for Javaiçin OSGi paketini kullanırsanız, OSGi gibi birden çok sınıf yükleyici ortamındaki sınıfların yüklenmesi sırasında oluşan bir sorun nedeniyle, Java ' ta yazılan kanal çıkış sınıfları desteklenmez. Bir kullanıcı paketi, IBM WebSphere MQ classes for Java paketinin farkında olabilir, ancak IBM WebSphere MQ classes for Java paketi herhangi bir kullanıcı kod paketinin farkında değildir. Sonuç olarak, bir IBM WebSphere MQ classes for Java kod paketinde kullanılan sınıf yükleyici, bir kullanıcı kod paketinde bulunan bir kanal çıkış sınıfını yükleyemez.

OSGi hakkında daha fazla bilgi için OSGi alliance web sitesine bakın.

IBM WebSphere MQ classes for Java yapılandırma dosyası

Bir IBM WebSphere MQ classes for Java yapılandırma dosyası, IBM WebSphere MQ classes for Javaürününü yapılandırmak için kullanılan özellikleri belirtir.

IBM WebSphere MQ classes for Java yapılandırma dosyasının biçimi, standart bir Java özellikler dosyasının biçimidir.

V 7.5.0.9 From IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 9, a sample configuration file that is called `mqjava.config` is supplied in the `bin` subdirectory of the IBM WebSphere MQ classes for Java installation directory. Bu dosya, desteklenen tüm özellikleri ve bunların varsayılan değerlerini belgeler.

Not: IBM WebSphere MQ kuruluşu ilerideki bir Düzeltme Paketine yükseltildiğinde örnek yapılanış kütüğünün üzerine yazılır. Bu nedenle, örnek yapılanış kütüğünün bir kopyasını uygulamalarınızla birlikte kullanmak için yapmanız önerilir.

Bir IBM WebSphere MQ classes for Java yapılandırma dosyasının adını ve konumunu seçebilirsiniz. Uygulamanıza başladığınızda, aşağıdaki biçimi kullanarak bir **java** komutu kullanın:

```
java -Dcom.ibm.msg.client.config.location=config_file_url application_name
```

Komutta *config_file_url* , IBM WebSphere MQ classes for Java yapılandırma dosyasının adını ve yerini belirten bir URL ' dir. Şu tiplerin URL ' leri desteklenir: `http`, `file`, `ftp`ve `jar`.

Aşağıdaki örnek bir **java** komutunu göstermektedir:

```
java -Dcom.ibm.msg.client.config.location=file:/D:/mydir/mqjava.config MyAppClass
```

Bu komut, yerel Windows sisteminde `D:\mydir\mqjava.config` dosyası olarak IBM WebSphere MQ classes for Java yapılandırma dosyasını tanımlar.

Bir uygulama ile kuyruk yöneticisi ya da aracı arasındaki desteklenen aktarımlardan herhangi biriyle bir IBM WebSphere MQ classes for Java yapılandırma dosyası kullanılabilir.

Bir IBM WebSphere MQ classes for Java yapılanış dosyasında belirtilen özelliklerin geçersiz kılınması

Bir IBM WebSphere MQ MQI client yapılandırma dosyası, IBM WebSphere MQ classes for Javauygulamasını yapılandırmak için kullanılan özellikleri de belirtebilir. Ancak, bir IBM WebSphere MQ MQI client yapılanış dosyasında belirtilen özellikler, yalnızca bir uygulama istemci kipinde bir kuyruk yöneticisine bağlandığında geçerlidir.

Gerekirse, bir IBM WebSphere MQ MQI client yapılandırma dosyasındaki herhangi bir özniteliği bir IBM WebSphere MQ classes for Java yapılandırma dosyasında bir özellik olarak belirterek geçersiz kılabilirsiniz. Bir IBM WebSphere MQ MQI client yapılandırma dosyasındaki bir özniteliği geçersiz kılmak için, IBM WebSphere MQ classes for Java yapılandırma dosyasında şu biçimdeki bir girişi kullanın:

```
com.ibm.mq.cfg.stanza.propName=propValue
```

Girdideki değişkenler aşağıdaki anlamlara sahiptir:

stanza
Özniteliği içeren IBM WebSphere MQ MQI client yapılandırma dosyasındaki stanza adının adı.

propName
IBM WebSphere MQ MQI client yapılandırma dosyasında belirtildiği gibi özniteliğin adı.

propValue
IBM WebSphere MQ MQI client yapılandırma dosyasında belirtilen özniteliğin değerini geçersiz kılan özelliğin değeri.

Diğer bir seçenek olarak, özelliği **java** komutunda bir sistem özelliği olarak belirterek bir IBM WebSphere MQ MQI client yapılanış kütüğündeki bir özniteliği geçersiz kılabilirsiniz. Özelliği bir sistem özelliği olarak belirtmek için önceki biçimi kullanın.

Bir IBM WebSphere MQ MQI client yapılandırma dosyasında yalnızca aşağıdaki öznitelikler IBM WebSphere MQ classes for Javaile ilgilidir. Diğer öznitelikleri belirtmiş ya da geçersiz kıyorsanız, bu herhangi bir etkisi olmaz. Specifically, note that the `ChannelDefinitionFile` and `ChannelDefinitionDirectory` in the İstemci yapılandırma dosyasının STANA kısmı are not used. CCDT ' yi IBM WebSphere MQ classes for Javaile nasıl kullanabilmeye ilişkin ayrıntılı bilgi için bkz. “Using a client channel definition table with IBM WebSphere MQ classes for Java” sayfa 645 .

Çizelge 86. İstemci yapılandırma dosyasında hangi özniteliğin bulunduğunu içeren

Stanza	**Öznitelik**
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultYolu
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultPath64
ClientExitİstemci yapılandırma dosyasının yol gösterişi	JavaExitsClasspath
İstemci yapılandırma dosyasınınMessageBuffer kısmı	MaximumSize

Çizelge 86. İstemci yapılandırma dosyasında hangi özniteliğin bulunduğunu içeren (devamı var)	
Stanza	**Öznitelik**
İstemci yapılandırma dosyasınınMessageBuffer kısmı	PurgeTime
İstemci yapılandırma dosyasınınMessageBuffer kısmı	UpdatePercentage
İstemci yapılandırma dosyasının TCP stanzası	ClntRcvBufSize
İstemci yapılandırma dosyasının TCP stanzası	ClntSndBufSize
İstemci yapılandırma dosyasının TCP stanzası	Bağlantı_Zamanaşımı
İstemci yapılandırma dosyasının TCP stanzası	KeepAlive

IBM WebSphere MQ MQI client yapılandırması hakkında daha fazla bilgi için bkz. Yapılandırma dosyası kullanarak istemci yapılandırılması.

İlgili görevler

IBM WebSphere MQ classes for Java uygulamalarının izlenmesi

Java Standart Ortam İzleme Stanzası

IBM WebSphere MQ classes for Java izleme olanağını yapılandırmak için Java Standard Environment Trace Settings stanza olanağını kullanabilirsiniz.

com.ibm.msg.client.commonservices.trace.outputName = `traceOutputName`

`traceOutputName` , izleme çıkışının gönderileceği dizin ve dosya adıdır.

İzleme dosyasının varsayılan adı, bir uygulama tarafından kullanılmakta olan IBM WebSphere MQ classes for Java sürümüne bağlıdır:

- For IBM WebSphere MQ classes for Java for Version 7.5.0, Fix Pack 8 or earlier, *traceOutputAd* defaults to a file named `mqjms_%PID%.trc` in the current working directory.
- V 7.5.0.9 IBM WebSphere MQ classes for Java 'tan Version 7.5.0, Fix Pack 9' dan, *traceOutputAdı* varsayılan olarak, yürürlükteki çalışma dizininde `mqjava_%PID%.trc` adlı bir dosyaya ayarlanır.

Burada *%PID%* geçerli işlem tanıtıcısıdır. If a process ID is unavailable, a random number is generated and prefixed with the letter `f`. İşlem tanıtıcısını belirttiğiniz bir dosya adına eklemek için `%PID%` dizgisini kullanın.

Başka bir dizin belirtirseniz, bu dizin var olmalıdır ve bu dizin için yazma izniniz olmalıdır. Yazma izniniz yoksa, izleme çıkışı `System.err`olarak yazılır.

com.ibm.msg.client.commonservices.trace.include = `includeList`

`includeList` , izlenecek paketlerin ve sınıfların bir listesidir ya da özel değerler `TüM` ya da `YOK`.

Paketi ya da sınıf adlarını noktalı virgül (`;`) ile ayırın. **`includeList`** varsayılan olarak `ALL`olarak ayarlanır ve IBM WebSphere MQ classes for Javaiçinde bulunan tüm paketleri ve sınıfları izler.

Not: Bir paket dahil edebilir, ancak o paketin alt paketlerini dışlayabilirsiniz. For example, if you include package `a.b` and exclude package `a.b.x`, the trace includes everything in `a.b.y` and `a.b.z`, but not `a.b.x` or `a.b.x.1`.

com.ibm.msg.client.commonservices.trace.exclude = `excludeList`

`excludeList` , izlenmeyen paketler ve sınıfların bir listesidir ya da özel değerler `TüM` ya da `YOK`.

Paketi ya da sınıf adlarını noktalı virgül (`;`) ile ayırın. **`excludeList`** varsayılan olarak `NONE`değerine ayarlanır ve bu nedenle, IBM WebSphere MQ classes for Java içindeki hiçbir paketi ve sınıfı izlenmekten çıkarır.

Not: Bir paketi dışlayabilir, ancak o paketin alt paketlerini de içerebilirsiniz. For example, if you exclude package `a.b` and include package `a.b.x`, the trace includes everything in `a.b.x` and `a.b.x.1`, but not `a.b.y` or `a.b.z`.

Hem içerilen hem de dışlanan paket ya da sınıf, aynı düzeyde belirtilmiş ve dışlanmış olarak dahil edilir.

com.ibm.msg.client.commonservices.trace.maxBytes = maxArrayBytes
maxArrayBytes , herhangi bir bayt dizisinden izlenen bayt sayısı üst sınışıdır.

maxArrayBytes pozitif bir tamsayıya ayarlandıysa, izleme dosyasına yazılacak bayt dizideki bayt sayısını sınırlar. *maxArrayBayt* yazıldıktan sonra bayt dizisini keser. **maxArrayBytes** ayarı sonucunda elde edilen izleme dosyası büyüklüğü azaltılıyor ve izleme işlemi, uygulamanın performansında etkisini azaltır.

Bu özellik için `0` değeri, izleme dosyasına herhangi bir byte dizisi içeriğinin hiçbirinin gönderilmemesinin anlamına gelir.

Varsayılan değer `-1`' dir; bu değer, izleme dosyasına gönderilen bayt dizesindeki bayt sayısındaki herhangi bir sınırı kaldırır.

com.ibm.msg.client.commonservices.trace.limit = maxTraceBytes
maxTraceBytes , bir izleme çıkış dosyasına yazılan bayt sayısı üst sınışıdır.

maxTraceBytes , **traceCycles**ile çalışır. Yazılan izleme baytlarının sayısı sınıra yakınsa, dosya kapatılır ve yeni bir izleme çıkış dosyası başlatılır.

`0` değeri, izleme çıkış dosyasının sıfır uzunluğuna sahip olduğu anlamına gelir. Varsayılan değer `-1`' dir. Bu, bir izleme çıkış dosyasına yazılacak veri miktarının sınırsız olduğu anlamına gelir.

com.ibm.msg.client.commonservices.trace.count = traceCycles
traceCycles , çevrim yoluyla çevrilecek izleme çıkış dosyalarının sayısıdır.

Yürürlükteki izleme çıkış dosyası **maxTraceBytes**ile belirtilen sınıra ulaşırsa, dosya kapatılır. Ek izleme çıkışı, sonraki izleme çıkış dosyasına sırayla yazılır. Her izleme çıkış dosyası, dosya adının sonuna eklenen sayısal bir sonek tarafından ayırt edilir. The current or most recent trace output file has the suffix `.trc.0`, the next most recent trace output file ends with `.trc.1`, and so on. Daha eski izleme dosyaları, sınırlamaya kadar aynı numaralandırma örüntülerini izler.

traceCycles varsayılan değeri 1' dir. **traceCycles** 1ise, yürürlükteki izleme çıkış dosyası büyüklük üst sınırına ulaştığında, dosya kapatılır ve silinir. Aynı adı taşıyan yeni bir izleme çıkış dosyası başlatılmış. Bu nedenle, bir kerede tek bir izleme çıkış dosyası vardır.

com.ibm.msg.client.commonservices.trace.parameter = traceParameters
traceParameters yöntem değiştirgelerinin ve dönüş değerlerinin izlemenin içerilip içerilmeyeceğini denetler.

traceParameters varsayılan olarak TRUEdeğerine ayarlanır. **traceParameters** FALSEolarak ayarlandıysa, yalnızca yöntem imzaları izlenir.

com.ibm.msg.client.commonservices.trace.compress = compressedTrace
İzleme çıkışını sıkıştırmak için **compressedTrace** değerini TRUE olarak ayarlayın.

compressedTrace varsayılan değeri FALSEdeğeridir.

compressedTrace değeri TRUEolarak ayarlanırsa, izleme çıkışı sıkıştırılır. Varsayılan izleme çıkışı dosyası adı, `.trz`uzantısına sahiptir. If compression is set to FALSE, the default value, the file has the extension `.trc` to indicate it is uncompressed. Ancak, izleme çıkışının dosya adı **traceOutputName** ' ta belirtilirse, bu ad kullanılır ve dosyaya sonek uygulanmaz.

Sıkıştırılmış izleme çıkışı, sıkıştırılmamış boyuttan küçük. Daha az G/Ç olduğundan, sıkıştırılmamış izlemenin daha hızlı bir şekilde yazılabiliyor. Sıkıştırılmış izleme, IBM WebSphere MQ classes for Java ' un performansını sıkıştırılmamış izlemekten daha az etkiye sahiptir.

maxTraceBytes ve **traceCycles** ayarlandıysa, birden çok düz dosya yerine birden çok sıkıştırılmış izleme dosyası oluşturulur.

IBM WebSphere MQ classes for Java , denetimli olmayan bir şekilde sona ererse, sıkıştırılmış izleme dosyası geçerli olmayabilir. Bu nedenle, izleme sıkıştırması yalnızca IBM WebSphere MQ classes for Java kapatıldığında denetimli bir biçimde kullanılmalıdır. İzleme sıkıştırmasını yalnızca araştırılmakta olan sorunlar JVM ' nin beklenmedik bir şekilde durmasına neden olmaz ise kullanın. `System.Halt()` kapatma ya da olağandışı olmayan, denetimsiz JVM sonlandırmalarıyla sonuçlanabilen sorunları tanılarken izleme sıkıştırması kullanmayın.

com.ibm.msg.client.commonservices.trace.level = `traceLevel`

`traceLevel` , izleme için bir süzgeç düzeyi belirtir. Tanımlanan izleme düzeyleri aşağıdaki gibidir:

Çizelge 87. Her izleme düzeyi için izlenecek öğe

Değer	İzlenecek öğe
0	İzleme kapatıldı
1	Kural Dışı Durumlar
3	Kural Dışı Durumlar Uyarılar
6	Kural Dışı Durumlar Uyarılar Bilgi izleme noktaları
8	Kural Dışı Durumlar Uyarılar Bilgi izleme noktaları Yöntem girişi ve çıkışı
9	Kural Dışı Durumlar Uyarılar Bilgi izleme noktaları Yöntem girişi ve çıkışı IBM WebSphere MQ classes for Java ile bir kuyruk yöneticisi arasında gönderilen veriler.

Not: IBM Desteği tarafından tersi belirtilmediği sürece her zaman 9 değerini kullanın.

IBM WebSphere MQ classes for Java ve yazılım yönetimi araçları

Apache Maven gibi yazılım yönetimi araçları IBM WebSphere MQ classes for Javaile birlikte kullanılabilir.

Birçok büyük geliştirme kuruluşu, bu araçları, üçüncü kişi kitaplıklarının havuzlarını merkezi olarak yönetmek için kullanır.

IBM WebSphere MQ classes for Java , JAR dosyalarından oluşan bir sayıdan oluşur. Bu API ' yı kullanarak Java dili uygulamaları geliştirirken, uygulamanın geliştirilmekte olduğu makinede bir IBM WebSphere MQ Server, IBM WebSphere MQ Client ya da IBM WebSphere MQ Client SupportPac kuruluşu gereklidir.

Bir yazılım yönetimi aracı kullanmak ve IBM WebSphere MQ classes for Java ' yi merkezi olarak yönetilen bir havuza oluşturan JAR dosyalarını eklemek istiyorsanız, aşağıdaki noktalar gözlenmelidir:

- Bir havuz ya da taşıyıcı, yalnızca kuruluşunuz içindeki geliştiriciler tarafından kullanılabilir hale getirilmelidir. Kuruluşun dışındaki herhangi bir dağılıma izin verilmez.
- Havuzun tek bir IBM WebSphere MQ yayınından ya da Düzeltme Paketinden eksiksiz ve tutarlı bir JAR dosyası kümesi içermesi gerekir.

- Havuzu, IBM Desteği tarafından sağlanan herhangi bir bakım ile güncellemekle göreviniz vardır.

IBM WebSphere MQ Version 7.5için, havuza aşağıdaki JAR dosyalarının kurulması gerekir:

- com.ibm.mq.commonservices.jar
- com.ibm.mq.jar
- com.ibm.mq.jmqi.jar
- com.ibm.mq.pcf.jar
- com.ibm.mq.headers.jar
- connector.jar

IBM WebSphere MQ uygulamaları için kuruluş sonrası ayarları

IBM WebSphere MQkurulduktan sonra, kendi uygulamalarınızı çalıştırmak için kuruluşunuzu yapılandırabilirsiniz.

Remember to check the IBM WebSphere MQ README file for later, or more specific, information for your environment.

Before attempting to run an IBM WebSphere MQ classes for Java application in bindings mode, make sure that you have configured IBM WebSphere MQ as described in Yapılandırılıyor .

Kuyruk yöneticinizin, Java için WebSphere MQ sınıflarından istemci bağlantılarını kabul edecek şekilde yapılandırılması

Kuyruk yöneticinizi istemcilerden gelen bağlantı isteklerini kabul edecek şekilde yapılandırmak için, bir sunucu bağlantı kanalının kullanımını tanımlayın ve bu kanaldan izin verin ve bir dinleyici programı başlatın.

Ayrıntılar için bkz. “Örnek programların hazırlanması ve çalıştırılması” sayfa 104.

Java Security Manager altında Java uygulamaları için WebSphere MQ sınıflarının çalıştırılması

Java için WebSphere MQ sınıfları Java Security Manager etkinleştirilmiş olarak çalıştırılabilir. Güvenlik Yöneticisi 'ni etkinleştiren uygulamaları başarıyla çalıştırmak için, Java sanal makinenizi (JVM) uygun bir ilke tanımlama dosyasıyla yapılandırmalısınız.

Bunun en basit yolu, JRE ' nizle birlikte sağlanan ilke dosyasını değiştirmemeniz. Çoğu sistemde, bu dosya JRE dizininize göre `lib/security/java.policy`yolunda saklanır. İlke dosyalarını tercih ettiğiniz düzenleyiciyi ya da JRE ' nizle birlikte sağlanan `policytool` programını kullanarak düzenleyebilirsiniz.

You must give authority to the com.ibm.mq.jmqi.jar file so that it can:

- Yuva yarat (istemci kipinde)
- Yerli kitaplığı yükle (bağ tanımları kipinde)
- Ortamdaki çeşitli özellikleri okuyun

The system property os.name must be available to the WebSphere MQ classes for Java when running under the Java Security Manager.

Aşağıda, Java için WebSphere MQ sınıflarının varsayılan güvenlik yöneticisi altında başarılı bir şekilde çalışmasını sağlayan bir ilke dosyası girişi örneği yer alıyor:

```
grant codeBase "file:MQ_INSTALLATION_PATH/java/lib/com.ibm.mq.jmqi.jar" {
  //Required
  permission java.util.PropertyPermission "user.name","read";
  permission java.util.PropertyPermission "os.name","read";
  //Required if mqclient.ini/mqs.ini configuration files are used
  permission java.io.FilePermission "/var/mqm/mqclient.ini","read";
  permission java.io.FilePermission "/var/mqm/mqs.ini","read";
  //For the client transport type.
  permission java.net.SocketPermission "*","connect";
  //For the bindings transport type.
  permission java.lang.RuntimePermission "loadLibrary.*";
  //For applications that use CCDT tables (access to the CCDT
```

```
AMQCLCHL.TAB)
  permission java.io.FilePermission
"/var/mqm/qmgrs/QMGR/@ipcc/AMQCLCHL.TAB","read";
  //For applications that use User Exits
  permission java.io.FilePermission "/var/mqm/exits/*","read";
  permission java.lang.RuntimePermission "createClassLoader";
  //Required for the z/OS platform
  permission java.util.PropertyPermission
"com.ibm.vm.bitmode","read";
};
grant codeBase
"file:MQ_INSTALLATION_PATH/java/lib/com.ibm.mq.commonservices.jar" {
  permission java.util.PropertyPermission "user.dir","read";
  permission java.util.PropertyPermission "line.separator","read";
  //tracing permissions
  permission java.util.PropertyPermission "com.ibm.mq.commonservices", "read";
  permission java.util.logging.LoggingPermission "control";
  //For access to the trace properties file.
  permission java.io.FilePermission "/tmp/trace.properties", "read";
  //For access to the trace output files.
  permission java.io.FilePermission "/tmp/*", "read,write";
};
```

Notlar:

- *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.
- Bir ilke dosyasının bu örneği, Java için WebSphere MQ sınıflarının güvenlik yöneticisi altında doğru şekilde çalışmasını sağlar; ancak, uygulamalarınız çalışmadan önce kendi kodunuzun doğru çalışması için kendi kodunuzu etkinleştirmeniz gerekebilir.
- Java için WebSphere MQ sınıflarının bir uygulamanın Java arşiv (JAR) dosyalarına erişmelerini sağlamak için, ilk `grant` deyime aşağıdaki izni ekleyin:

  ```
  permission java.io.FilePermission "/path_to_your_app/-", "read";
  ```

- Bu `grant` deyimlerini ilke yapılandırma dosyanızın içinde kullanmak için, Java için WebSphere MQ sınıflarının kurulu olduğu yere ve uygulamalarınızı sakladığınız yere bağlı olarak yol adlarını değiştirmeniz gerekebilir.
- Java için WebSphere MQ sınıflarıyla verilen örnek kod, güvenlik yöneticisiyle birlikte kullanılmak üzere özel olarak etkinleştirilmemiş; ancak IVT sınamaları bu ilke dosyasıyla ve varsayılan güvenlik yöneticisiyle çalışır.

Java kuruluşuna ilişkin IBM WebSphere MQ sınıflarının doğrulanması

Bir kuruluş doğrulama programı (MQIVP), Javaiçin IBM WebSphere MQ sınıflarıyla birlikte sağlanır. Bu programı, Javaiçin IBM WebSphere MQ sınıflarının tüm bağlantı kiplerini sınamak için kullanabilirsiniz.

Program, doğrulamak istediğiniz bağlantı kipini belirlemek için bir dizi seçenek ve diğer veri bilgi istemlerini içerir. Kuruluşunuzu doğrulamak için aşağıdaki yordamı kullanın:

1. Programı istemci kipinde çalıştırabildiyseniz, kuyruk yöneticinizi “Örnek programların hazırlanması ve çalıştırılması” sayfa 104içinde açıklandığı gibi yapılandırın. Kullanılacak kuyruk SYSTEM.DEFAULT.LOCAL.QUEUE.
2. Programı istemci kipinde çalıştırabildiyseniz, bkz. “Java için WebSphere MQ sınıflarının kullanılması” sayfa 627.

   Bu yordamın geri kalan adımlarını, programı çalıştırabilmekte olduğunuz sistemde gerçekleştirin.
3. CLASSPATH ortam değişkeninizi, “Java için WebSphere MQ sınıflarıyla ilgili ortam değişkenleri” sayfa 631içindeki yönergelere göre güncellediğinizden emin olun.
4. Dizini `MQ_INSTALLATION_PATH/mqm/VRM/java/samples/wmqjava`olarak değiştirin; burada MQ_INSTALLATION_PATH, IBM WebSphere MQ kurulumunuzun yolu ve VRM , ürünün sürüm, yayın ve değişiklik numarasıdır. Daha sonra komut istemine şunu girin:

   ```
   java -Djava.library.path=library_path MQIVP
   ```

Burada *kitaplık_yolu* , Java kitaplıkları için IBM WebSphere MQ sınıflarının yoludur (bkz. Java kitaplıkları için WebSphere MQ sınıfları).

(1) imlenmiş bilgi isteminde aşağıdaki bilgiler yer aldı:

- TCP/IP bağlantısı kullanmak için bir IBM WebSphere MQ sunucusu anasistem adı girin.
- Yerel bağlantıyı kullanmak için (bağ tanımları kipi) alanı boş bırakın (ad girmeyin).

Program aşağıdakileri yapmaya çalışır:

- 1. Kuyruk yöneticisine bağlan
- 2. SYSTEM.DEFAULT.LOCAL.QUEUEkuyruğunu açın, kuyruğa bir ileti yazın, kuyruktan bir ileti alın ve kuyruğu kapatın.
- 3. Kuyruk yöneticisinden bağlantı kesme
- 4. İşlemler başarılı olursa bir ileti döndür

Burada, görebileceğiniz bilgi istemlerinin ve yanıtların bir örneği yer alır. Gerçek bilgi istemleri ve yanıtlarınız IBM WebSphere MQ ağınıza bağlıdır.

```
Please enter the IP address of the MQ server              : ipaddress(1)
Please enter the port to connect to                       : (1414)(2)
Please enter the server connection channel name            : channelname(2)
Please enter the queue manager name                       : qmname
Success: Connected to queue manager.
Success: Opened SYSTEM.DEFAULT.LOCAL.QUEUE
Success: Put a message to SYSTEM.DEFAULT.LOCAL.QUEUE
Success: Got a message from SYSTEM.DEFAULT.LOCAL.QUEUE
Success: Closed SYSTEM.DEFAULT.LOCAL.QUEUE
Success: Disconnected from queue manager


Tests complete -
SUCCESS: This MQ Transport is functioning correctly.
Press Enter to continue ...
```

Not:

1. Sunucu bağlantısını seçerseniz, (2)ile işaretlenen bilgi istemlerini görmeyin.

IBM WebSphere MQ sorunlarının çözülmesi

Başlangıçta kuruluş doğrulama programını çalıştırın. Ayrıca izleme tesisini de kullanmak zorunda kalabilirsiniz.

Bir program başarıyla tamamlanmazsa, kuruluş doğrulama programını çalıştırın ve tanılama iletilerinde verilen öneriyi izleyin. Bu program “Java kuruluşuna ilişkin IBM WebSphere MQ sınıflarının doğrulanması” sayfa 639içinde açıklanmaktadır.

Sorunlar devam ederse ve IBM hizmet ekibiyle iletişim kurmanız gerekiyorsa, izleme olanağını açmanız istenebilir. Bu işlemi aşağıdaki örnekte gösterildiği gibi yapın.

`MQIVP` programını izlemek için:

- Bir *com.ibm.mq.commonservices* özellikler dosyası oluşturun (bkz. com.ibm.mq.commonserviceskomutunu kullanma .
- Aşağıdaki komutu girin:

```
java -Dcom.ibm.mq.commonservices=commonservices_properties_file java
-Djava.library.path=library_path MQIVP -trace
```

Burada:

- *commonservice_properties_file* , *com.ibm.mq.commonservices* özellikler (properties) dosyasına giden yoldur (dosya adı da içinde olmak üzere).
- *kitaplık_yolu* , Java kitaplıkları için WebSphere MQ sınıflarının yoludur (bkz. Java kitaplıkları için WebSphere MQ sınıfları).

İzlemenin nasıl kullanılacağı hakkında daha fazla bilgi için IBM WebSphere MQ classes for Java uygulamalarını izlemebaşlıklı konuya bakın.

Programcılar için giriş

Bu konu grubu, programcılara ilişkin genel bilgileri içerir.

Program yazma hakkında daha ayrıntılı bilgi için bkz. “Java uygulamaları için WebSphere MQ sınıflarının yazılması” sayfa 641.

Java arabirimi için WebSphere MQ sınıfları

Yordamsal WebSphere MQ uygulama programlama arabirimi, nesnelerle hareket eden fiilleri kullanır. Java programlama arabirimi, yöntemleri çağırarak üzerinde işlem yapmak için nesneleri kullanır.

Yordamsal WebSphere MQ uygulama programlama arabirimi, aşağıdaki gibi fiiller çevresinde oluşturulur:

```
MQBACK, MQBEGIN, MQCLOSE, MQCONN, MQDISC,
MQGET, MQINQ, MQOPEN, MQPUT, MQSET, MQSUB
```

Bu fiillerin tümü, bir parametre olarak, çalışacakları WebSphere MQ nesnesi için bir tanıtıcı olarak alır. Java nesne yönelimli olduğu için, Java programlama arabirimi bu tura döner. Programınız, bu nesnelerle ilgili yöntemler çağırarak işlem yapmak istediğiniz bir WebSphere MQ nesnelerinden oluşur.

Yordamsal arabirimi kullanırken, MQDISC çağrısını (Hconn, CompCode, Reason) kullanarak bir kuyruk yöneticisinden bağlantınız kesilir; burada *Hconn* , kuyruk yöneticisinin tanıtıcısıdır.

Java arabiriminde, kuyruk yöneticisi, MQQueueManagersınıfı bir nesle temsil edilir. Bu sınıftaki bağlantı kesme () yöntemini çağırarak, kuyruk yöneticisinden bağlantıyı kesmenizi sağlar.

```
// declare an object of type queue manager
MQQueueManager queueManager=new MQQueueManager();
...
// do something...
...
// disconnect from the queue manager
queueManager.disconnect();
```

Java uygulamaları için WebSphere MQ sınıflarının yazılması

Bu konu derlemi, Java uygulamalarının WebSphere MQ sistemleriyle etkileşimde bulunabilmelerine yardımcı olacak bilgiler sağlar.

Java 'nın WebSphere MQ kuyruklarına erişmek üzere WebSphere MQ sınıflarını kullanmak için, iletileri içeren çağrıları içeren Java uygulamalarını ve WebSphere MQ kuyruklarından ileti almak için kullanılır. Tek tek sınıflara ilişkin ayrıntılar için WebSphere MQ class for Javabaşlıklı konuya bakın.

Not: Otomatik istemci yeniden bağlantısı, Java için WebSphere MQ sınıfları tarafından desteklenmez.

Java bağlantı kipleri içinWebSphere MQ sınıfları

Java için WebSphere MQ sınıflarına ilişkin programınız, kullanmak istediğiniz bağlantı kiplerine bazı bağımlılıklar içerir.

İstemci bağlantıları kullanıyorsanız, IBM WebSphere MQ MQI client ' den bir dizi fark vardır; ancak kavramsal olarak benzerdir. Bağ tanımları kipini kullanıyorsanız, fastpath bağ tanımlarını kullanabilir ve MQBEGIN komutunu verebilirsiniz. MQEnvironment sınıfındaki değişkenleri ayarlayarak hangi kipin kullanılacağını belirtmenizi sağlar.

Java istemcisi bağlantıları için WebSphere MQ sınıfları

When WebSphere MQ classes for Java is used as a client, it is like the IBM WebSphere MQ MQI client, but has a number of differences.

Java için WebSphere MQ sınıfları için istemci olarak kullanılmak üzere programlıyorsanız, aşağıdaki farklardan haberdar olun:

- Yalnızca TCP/IP ' yi destekler.
- Başlatma sırasında herhangi bir WebSphere MQ ortam değişkeni okumaz.
- Bir kanal tanımlamasında ve ortam değişkenlerinde saklanacak bilgiler, Ortam adlı bir sınıfa saklanabilir. Diğer bir seçenek olarak, bu bilgiler, bağlantı yapıldığında parametre olarak geçirilebilir.
- Hata ve kural dışı durum koşulları, `MQException` sınıfında belirtilen günlüğe yazılır. Varsayılan hata hedefi, Java konsoldur.
- Bir WebSphere MQ istemcisi yapılandırma dosyasındaki yalnızca aşağıdaki öznitelikler, Java için WebSphere MQ sınıflarıyla ilgilidir. Diğer öznitelikleri belirtirseniz, bunlar geçersiz olur.

Stanza	**Öznitelik**
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultYolu
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultPath64
ClientExitİstemci yapılandırma dosyasının yol gösterişi	JavaExitsClasspath
İstemci yapılandırma dosyasınınMessageBuffer kısmı	MaximumSize
İstemci yapılandırma dosyasınınMessageBuffer kısmı	PurgeTime
İstemci yapılandırma dosyasınınMessageBuffer kısmı	UpdatePercentage
İstemci yapılandırma dosyasının TCP stanzası	ClntRcvBufSize
İstemci yapılandırma dosyasının TCP stanzası	ClntSndBufSize
İstemci yapılandırma dosyasının TCP stanzası	Bağlantı_Zamanaşımı
İstemci yapılandırma dosyasının TCP stanzası	KeepAlive

- If connecting to a queue manager that requires character data to be converted, then the V7 Java client is now capable of doing the conversion if queue manager is unable to do so. İstemci JVM ' nin, istemcinin CCSID 'si ile kuyruk yöneticisinin arasındaki dönüştürmeyi desteklemesi gerekir.
- Otomatik istemci yeniden bağlanması, Java için WebSphere MQ sınıfları tarafından desteklenmez.

İstemci kipinde kullanıldığında, *WebSphere MQ class for Java* , MQBEGIN çağrısını desteklemez.

Desteklenen ortamlarla ilgili ek bilgi için “Java için WebSphere MQ sınıflarına ilişkin bağlantı seçenekleri” sayfa 628 ' e bakın.

Java bağ tanımları kipi için WebSphere MQ sınıfları

Java için WebSphere MQ sınıflarının bağ tanımları kipi, istemci kipinden üç ana yol kadar farklıdır.

Bağ tanımları kipinde kullanıldığında, Java için WebSphere MQ sınıfları Java Native Interface 'i (JNI), bir ağ üzerinden iletişim kurmak yerine, doğrudan var olan kuyruk yöneticisi API ' sına çağrılacak şekilde kullanır.

Varsayılan olarak, bağ tanımları kipinde Java için WebSphere MQ sınıflarını kullanan uygulamalar, *ConnectOption*, MQCNO_STANDARD_BAĞ tanımlarını kullanarak bir kuyruk yöneticisine bağlanır.

Java için WebSphere MQ sınıfları aşağıdaki *ConnectOptions*sınıflarını destekler:

- MQCNO_FASTPATH_BINDING
- MQCNO_STANDARD_BINDING

- MQCNO_SHARED_BINDING
- MQCNO_ISOLATED_BINDING

ConnectOptions(Bağlantı Seçenekleri) konusuna ilişkin ek bilgi için bkz. “MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 200.

Bindings mode supports the MQBEGIN call to initiate global units of work that are coordinated by the queue manager, on all platforms apart from WebSphere MQ for IBM i and WebSphere MQ for z/OS.

`MQEnvironment` sınıfı tarafından sağlanan parametrelerin çoğu bağ tanımları kipiyle ilgili değildir ve yoksayılır.

Desteklenen ortamlarla ilgili ek bilgi için “Java için WebSphere MQ sınıflarına ilişkin bağlantı seçenekleri” sayfa 628 ' e bakın.

Kullanılacak Java bağlantısı için hangi WebSphere MQ sınıflarının kullanılacağını tanımlama

Kullanılacak bağlantı tipi, MQEnvironment sınıfındaki değişkenlerin ayarlarıyla belirlenir.

İki değişken kullanılır:

MQEnvironment.properties
Bağlantı tipi, `CMQC.TRANSPORT_PROPERTY`anahtar adıyla ilişkili değer tarafından belirlenir. Olası değerler aşağıdaki gibidir:

CMQC.TRANSPORT_MQSERIES_BINDINGS
Bağ tanımları moduna bağlan

CMQC.TRANSPORT_MQSERIES_CLIENT
İstemci kipinde bağlan

CMQC.TRANSPORT_MQSERIES
Bağlantı kipi, *hostname* özelliğinin değerine göre belirlenir.

MQEnvironment.hostname
Bu değişkenin değerini aşağıdaki gibi ayarlayın:

- İstemci bağlantıları için, bu değişkenin değerini, bağlanmak istediğiniz IBM WebSphere MQ sunucusunun ana makine adına ayarlayın.
- Bağ tanımları kipi için bu değişkeni ayarlamayın ya da boş değere ayarlayın

Kuyruk yöneticilerine ilişkin işlemler

Bu konular derlemi, Java için WebSphere MQ sınıflarını kullanan bir kuyruk yöneticisinin nasıl bağlanacağını ve bağlantı kesileceğini açıklar.

Java için WebSphere MQ sınıfları için WebSphere MQ ortamını ayarlama

Bir uygulamanın istemci kipinde bir kuyruk yöneticisine bağlanmasını sağlamak için, uygulamanın kanal adını, anasistem adını ve kapı numarasını belirtmesi gerekir.

Not: Bu konudaki bilgiler, uygulamanızın istemci kipindeki bir kuyruk yöneticisine bağlanması durumunda geçerlidir. Bağ tanımları kipine bağlıysa, bu *değildir* . Bkz. “JMS için WebSphere MQ sınıflarına ilişkin bağlantı kipleri” sayfa 741

Kanal adını, anasistem adını ve kapı numarasını, MQEnvironment sınıfındaki alanlar olarak ya da MQQueueManager nesnesinin özellikleri olarak belirleyebilirsiniz.

MQEnvironment sınıfındaki alanları ayarlıyorsanız, bunlar, bir özellikler HASH çizelgesi tarafından geçersiz kılınan durumlar dışında, tüm uygulamanıza uygulanır. MQEnvironment 'da kanal adını ve anasistem adını belirtmek için aşağıdaki kodu kullanın:

```
MQEnvironment.hostname = "host.domain.com";
MQEnvironment.channel  = "java.client.channel";
```

Bu, bir **MQSERVER** ortam değişkeni ayarlamaya eşdeğerdir:

```
"java.client.channel/TCP/host.domain.com".
```

Varsayılan olarak, Java istemcileri 1414kapısındaki WebSphere MQ dinleyicisine bağlanmayı dener. Farklı bir kapı belirtmek için aşağıdaki kodu kullanın:

```
MQEnvironment.port = nnnn;
```

burada nnnn gerekli kapı numarasıdır

Özellikleri, yaratılışındaki bir kuyruk yöneticisi nesnesine geçerseniz, bu nesne yalnızca o kuyruk yöneticisine uygulanır. Create entries in a Hashtable object with keys of **hostname**, **channel**, and, optionally, **port**, and with appropriate values. Varsayılan kapıyı kullanmak için 1414, **port** girdisini atlayabilirsiniz. Özellikler karma çizelgesini kabul eden bir oluşturucu kullanarak MQQueueManager nesnesini yaratın.

Bir uygulama adı belirleyerek kuyruk yöneticiyle bağlantı tanımlanması

Bir uygulama, kuyruk yöneticisiyle olan bağlantısını tanıtan bir ad ayarlayabilir. Bu uygulama adı **DISPLAY CONN MQSC/PCF** komutu (alanın adı **APPLTAG**olarak adlandırılır) ya da WebSphere MQ Explorer **Application Connections** (Uygulama Bağlantıları) görüntülerinde gösterilir (burada alan adı **App name**olur).

Uygulama adları 28 karakterle sınırlıdır ve daha uzun adlar sığabilmek için kesilir. Bir uygulama adı belirtilmediyse, varsayılan değer olarak bir varsayılan değer sağlanır. Varsayılan ad, çağıran (ana) sınıfa dayalıdır; ancak, bu bilgi kullanılamıyorsa, `WebSphere MQ Client for Java` metni kullanılır.

Çağıran sınıfın adı kullanılırsa, gerekiyorsa, önde gelen paket adları kaldırılarak sığaca ayarlanır. Örneğin, çağrılan sınıf `com.example.MainAppise`, tam ad kullanılır, ancak çağıran sınıf `com.example.dictionaryAndThesaurus.multilingual.mainAppise`, bu ad, sınıf adının en uzun birleşimi ve kullanılabilir uzunluğa uyan en sağdaki paket adının en uzun birleşimi olduğu için `multilingual.mainApp` kullanılır.

Sınıf adının kendisi 28 karakterden uzunsa, sığaca kesilir. Örneğin, `com.example.mainApplicationForSecondTestCase` , `mainApplicationForSecondTest`olur.

Not: z/OS platformlarında çalışan kuyruk yöneticileri, uygulama adlarını belirlemeyi desteklemez.

MQEnvironment sınıfında bir uygulama adı belirlemek için, adı aşağıdaki kodu kullanarak **MQConstants.APPNAME_PROPERTY**anahtisiyle MQEnvironment.properties hash çizelgesine ekleyin:

```
MQEnvironment.properties.put(MQConstants.APPNAME_PROPERTY, "my_application_name");
```

To set an application name in the properties hash table that is passed to the MQQueueManager constructor, add the name to the properties hash table with a key of **MQConstants.APPNAME_PROPERTY**.

Bir WebSphere MQ istemcisi yapılanış kütüğünde belirtilen özelliklerin geçersiz kılınması

Bir WebSphere MQ istemcisi yapılanış dosyası, Java için WebSphere MQ sınıflarını yapılandırmak için kullanılan özellikleri de belirtebilir. Ancak, bir WebSphere MQ MQI istemcisi yapılanış dosyasında belirtilen özellikler yalnızca, bir uygulama istemci kipinde bir kuyruk yöneticisine bağlandığında geçerlidir.

If required, you can override any attribute in a WebSphere MQ configuration file in any of the following ways. Seçenekler öncelik sırasına göre gösterilir.

- Yapılandırma özelliği için bir Java sistem özelliği ayarlayın.
- Set the property in the MQEnvironment.properties map.
- Java5 ve sonraki yayın düzeylerinde, bir sistem ortam değişkeni ayarlayın.

Bir WebSphere MQ istemcisi yapılanış kütüğünde yalnızca aşağıdaki öznitelikler, Java için WebSphere MQ sınıflarıyla ilgilidir. Diğer öznitelikleri belirtmiş ya da geçersiz kıyorsanız, bu herhangi bir etkisi olmaz.

Stanza	Öznitelik
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultYolu
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultPath64
ClientExitİstemci yapılandırma dosyasının yol gösterişi	JavaExitsClasspath
İstemci yapılandırma dosyasınınMessageBuffer kısmı	MaximumSize
İstemci yapılandırma dosyasınınMessageBuffer kısmı	PurgeTime
İstemci yapılandırma dosyasınınMessageBuffer kısmı	UpdatePercentage
İstemci yapılandırma dosyasının TCP stanzası	ClntRcvBufSize
İstemci yapılandırma dosyasının TCP stanzası	ClntSndBufSize
İstemci yapılandırma dosyasının TCP stanzası	Bağlantı_Zamanaşımı
İstemci yapılandırma dosyasının TCP stanzası	KeepAlive

Java için WebSphere MQ sınıflarında bir kuyruk yöneticisine bağlanılması

MQQueueManager sınıfının yeni bir örneğini oluşturarak bir kuyruk yöneticisine bağlanın. Bağlantı kesme () yöntemini çağırarak, kuyruk yöneticisinden bağlantıyı kesin.

Artık MQQueueManager sınıfının yeni bir örneğini oluşturarak bir kuyruk yöneticisine bağlanmaya hazırsınız:

```
MQQueueManager queueManager = new MQQueueManager("qMgrName");
```

Kuyruk yöneticisinden bağlantıyı kesmek için, kuyruk yöneticisininde bağlantı kesme () yöntemini çağırın:

```
queueManager.disconnect();
```

Bağlantı kesme yöntemini çağırırsanız, o kuyruk yöneticisi aracılığıyla eriştiğiniz tüm açık kuyruklar ve işlemler kapatılır. Ancak, bu kaynakları kullanmayı bitirdiğinizde, bu kaynakları açık bir şekilde kapatmak için iyi bir programlama uygulamasıdır. Bunu yapmak için, ilgili nesnelerle ilgili close () yöntemini kullanın.

Kuyruk yöneticisiyle ilgili kesinleştirme () ve backout () yöntemleri, yordamsal arabirimle birlikte kullanılan MQCMIT ve MQBACK çağrılarına eşdeğerdir.

Using a client channel definition table with IBM WebSphere MQ classes for Java

Bir IBM WebSphere MQ classes for Java istemci uygulaması, istemci kanal tanımlama çizelgesinde (CCDT) saklanan istemci bağlantı kanalı tanımlarını kullanabilir.

As an alternative to creating a client connection channel definition by setting certain fields and environment properties in the `MQEnvironment` class or passing them to an `MQQueueManager` in a properties hash table, a IBM WebSphere MQ classes for Java client application can use client connection channel definitions that are stored in a client channel definition table. Bu tanımlar,IBM WebSphere MQ Script (MQSC) komutları ya da IBM WebSphere MQ Programmable Command Format (PCF) komutları ya da IBM WebSphere MQ Explorerkullanılarak yaratılır.

Uygulama bir MQQueueManager nesnesi yarattığında, IBM WebSphere MQ classes for Java istemcisi, uygun bir istemci bağlantısı kanal tanımlaması için istemci kanal tanımlama çizelgesini arar ve bir MQI kanalını başlatmak için kanal tanımlamasını kullanır. İstemci kanal tanımlama tabloları ve nasıl oluşturulacağı hakkında daha fazla bilgi için bkz. İstemci kanal tanımlama tablosu.

Bir istemci kanal tanımlama çizelgesi kullanmak için, uygulamanın önce bir URL nesnesi yaratması gerekir. URL nesnesi, istemci kanal tanımlama çizelgesini içeren dosyanın adını ve yerini tanımlayan ve dosyanın nasıl erişilebileceğini belirten bir URL adresi (uniform resource locator; URL) yerleştirir.

Örneğin, ccdt1.tab dosyası bir istemci kanal tanımlama çizelgesi içeriyorsa ve uygulamanın çalıştığı sistemde saklandıysa, uygulama aşağıdaki şekilde bir URL nesnesi yaratabilir:

```
java.net.URL chanTab1 = new URL("file:///home/admdata/ccdt1.tab");
```

Başka bir örnek olarak, ccdt2.tab dosyasının bir istemci kanal tanımlama çizelgesi içerdiğini ve uygulamanın çalışmakta olduğu sistemden farklı bir sistemde saklandığı varsayıldığını varsayalım. Dosyaya FTP iletişim kuralı kullanılarak erişilebildiyse, uygulama aşağıdaki şekilde bir URL nesnesi yaratabilir:

```
java.net.URL chanTab2 = new URL("ftp://ftp.server/admdata/ccdt2.tab");
```

Uygulama bir URL nesnesi yarattıktan sonra, uygulama, bir URL nesnesini parametre olarak alan oluşturuculardan birini kullanarak bir MQQueueManager nesnesi yaratabilir. Örnek:

```
MQQueueManager mars = new MQQueueManager("MARS", chanTab2);
```

Bu deyim, IBM WebSphere MQ classes for Java istemcisinin chanTab2URL nesnesi tarafından tanımlanan istemci kanal tanımlama çizelgesine erişmesini sağlar, uygun bir istemci bağlantısı kanalı tanımlaması için tabloyu arar ve sonra MARS adlı kuyruk yöneticisine bir MQI kanalı başlatmak için kanal tanımlamasını kullanır.

Bir uygulama istemci kanal tanımlama çizelgesi kullanıyorsa, aşağıdaki noktalara dikkat edin:

- Uygulama, parametre olarak bir URL nesnesi alan bir oluşturucu kullanan bir MQQueueManager nesnesi yarattığında, MQEnvironment sınıfında bir alan olarak ya da bir ortam özelliği olarak herhangi bir kanal adı ayarlanmamalıdır. Bir kanal adı ayarlandıysa, IBM WebSphere MQ classes for Java istemcisi bir MQExceptionatar. Kanal adını belirten alan ya da ortam özelliği, değeri boş değerli, boş bir dizgi ya da tüm boşluk karakterleri içeren bir dizgi ise, bu alan ya da ortam özelliğinin ayarlanabileceği varsayılır.
- MQQueueManager oluşturucudaki **queueManagerName** parametresi aşağıdaki değerlerden birine sahip olabilir:
  - Kuyruk yöneticisinin adı
  - Bir yıldız işareti (*) ve ardından bir kuyruk yöneticisi grubu adı gelir.
  - Yıldız işareti (*)
  - Boş değerli, boş bir dizgi ya da tüm boşluk karakterlerini içeren bir dizgi

  Bu değerler, Message Queue Interface (MQI) kullanan bir istemci uygulaması tarafından yayınlanan bir MQCONN çağrısındaki **QMgrName** parametresi için kullanılabilecek değerlerdir. Bu değerlerin anlamı hakkında daha fazla bilgi için bkz. “Message Queue Interface-Genel Bakış” sayfa 187.

  Uygulamanız bağlantı havuzlama kullanıyorsa, bkz. “Java için WebSphere MQ sınıflarında varsayılan bağlantı havuzunu denetleme” sayfa 666.
- IBM WebSphere MQ classes for Java istemcisi, istemci kanal tanımlama çizelgesinde uygun bir istemci bağlantısı kanalı tanımlaması bulduğunda, bu kanal tanımlamasından alınan bilgileri yalnızca bir MQI kanalı başlatmak için kullanır. Uygulamanın MQEnvironment sınıfında ayarmış olabileceği kanallarla ilgili alanlar ya da ortam özellikleri yoksayılır.

  Güvenli Yuva Arabirimi Katmanı (SSL) kullanıyorsanız, özellikle aşağıdaki noktalara dikkat edin:

– Bir MQI kanalı, yalnızca istemci kanal tanımlama çizelgesinden alınan kanal tanımlaması, IBM WebSphere MQ classes for Java tarafından desteklenen bir CipherSpec ' in adını belirtiyorsa SSL kullanır.
– İstemci kanalı tanımlama çizelgesi, sertifika iptal listelerini (CRL ' ler) bulunduran LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) sunucularının konularıyla ilgili bilgileri de içerir. IBM WebSphere MQ classes for Java istemcisi, CRL ' leri tutan LDAP sunucularına erişmek için yalnızca bu bilgileri kullanır.
– Bir istemci kanal tanımlama çizelgesi, OCSP yanıtlayıcıya ait bir yeri de içerebilir. IBM WebSphere MQ classes for Java , bir istemci kanal tanımlama çizelgesi dosyasında OCSP bilgilerini kullanamaz. Ancak, OCSP ' yi Using Online Certificate Protocol(Çevrimiçi Sertifika İletişim Kuralını Kullanma) bölümünde açıklandığı gibi yapılandırabilirsiniz.

İstemci kanal tanımlama çizelgesiyle SSL ' nin kullanılmasına ilişkin ek bilgi için, Bir MQI kanalının SSL kullandığını belirtmebaşlıklı konuya bakın.

Kanal çıkışlarını kullanıyorsanız aşağıdaki noktalara da dikkat edin:

– Bir MQI kanalı, kanal çıkışlarını ve diğer yöntemler kullanılarak belirlenen verileri kullanarak, istemci kanalı tanımlama çizelgesinden alınan kanal tanımlaması tarafından belirlenen kanal çıkışlarını ve ilişkili kullanıcı verilerini kullanır.
– İstemci kanalı tanımlama çizelgesinden alınan bir kanal tanımlaması, Java, C ya da C + + içinde yazılan kanal çıkışlarını belirtebilir. Java' ta kanal çıkışı nasıl yazılabileceğiyle ilgili daha fazla bilgi için bkz. “Java için WebSphere MQ sınıflarında kanal çıkışı yaratılması” sayfa 659. Bir kanal çıkışısının diğer dillerde nasıl yazılabileceği hakkında daha fazla bilgi için bkz. “Java için WebSphere MQ sınıflarıyla Java 'da yazılmamış kanal çıkışlarını kullanma” sayfa 663.

IBM WebSphere MQ classes for Java istemci bağlantıları için bir kapı aralığı belirtme

Bir uygulamanın iki yönlü olarak bağlanabileceği bir kapı ya da kapı aralığı belirtebilirsiniz.

Bir IBM WebSphere MQ classes for Java uygulaması istemci kipinde bir IBM WebSphere MQ kuyruk yöneticisine bağlanmayı denediğinde, bir güvenlik duvarı yalnızca belirtilen kapılardan ya da kapı aralığından kaynaklanan bağlantılara izin verebilir. Bu durumda, uygulamanın bağlanabileceği bir kapı ya da kapı aralığı belirtebilirsiniz. Kapı (lar) ı aşağıdaki şekillerde belirtebilirsiniz:

- MQEnvironment sınıfındaki localAddressSetting alanını ayarlayabilirsiniz. Örnek:

```
MQEnvironment.localAddressSetting = "192.0.2.0(2000,3000)";
```

- CMQC.LOCAL_ADDRESS_PROPERTY. Örnek:

```
(MQEnvironment.properties).put(CMQC.LOCAL_ADDRESS_PROPERTY,
                               "192.0.2.0(2000,3000)");
```

- MQQueueManager nesnesini oluşturabildiğinizde, "192.0.2.0(2000,3000)" değerine sahip LOCAL_ADDRESS_PROPERTY içeren bir özellikler hashtable öğesini geçirebilirsiniz.

Bu örneklerin her birinde, uygulama daha sonra bir kuyruk yöneticisine bağlandığında, uygulama, 192.0.2.0(2000)- 192.0.2.0(3000) aralığındaki yerel bir IP adresine ve kapı numarasına bağlanır.

Birden çok ağ arabirimine sahip bir sistemde, localAddress(yerel adres) ayar alanını ya da CMQC.LOCAL_ADDRESS_PROPERTY, bir bağlantı için hangi ağ arabiriminin kullanılması gerektiğini belirtmek için.

Kapı aralığını sınırladığınızda bağlantı hataları ortaya çıkabilir. Bir hata oluşursa, IBM WebSphere MQ neden kodu MQRC_Q_MGR_NOT_AVALABILIR ve şu iletiyi içeren bir MQException yayınlanır:

```
Socket connection attempt refused due to LOCAL_ADDRESS_PROPERTY restrictions
```

Belirtilen aralıktaki tüm kapılar kullanımdaysa ya da belirtilen IP adresi, anasistem adı ya da kapı numarası geçerli değilse (örneğin, negatif bir kapı numarası) bir hata oluşabilir.

Java için WebSphere MQ sınıflarındaki kuyruklara, konulara ve süreçlere erişilmesi

Kuyruklara, konulara ve süreçlere erişmek için MQQueueManager sınıfının yöntemlerini kullanın. MQOD (nesne tanımlayıcı yapısı), bu yöntemlerin parametrelerine daraltılır.

Kuyruklar

Bir kuyruğu açmak için, MQQueueManager sınıfının accessQueue yöntemini kullanabilirsiniz. Örneğin, queueManageradlı bir kuyruk yöneticisininde aşağıdaki kodu kullanın:

```
MQQueue queue = queueManager.accessQueue("qName",CMQC.MQOO_OUTPUT);
```

accessQueue yöntemi, MQQueue sınıfını yeni bir nesne döndürür.

Kuyruğu kullanmayı bitirdiğinizde, aşağıdaki örnekteki gibi kapatmak için close () yöntemini kullanın:

```
queue.close();
```

Ayrıca, MQQueue oluşturucusunu kullanarak bir kuyruk da yaratabilirsiniz. Parametreler, bir kuyruk yöneticisi değiştirgesinin eklenmesiyle tam olarak accessQueue yöntemi ile aynıdır. Örneğin:

```
MQQueue queue = new MQQueue(queueManager,
                            "qName",
                            CMQC.MQOO_OUTPUT,
                            "qMgrName",
                            "dynamicQName",
                            "altUserID");
```

Kuyruklar yaratırken bir dizi seçenek belirleyebilirsiniz. Bunlara ilişkin ayrıntılar için Class.com.ibm.mq.MQQueuebaşlıklı konuya bakın. Bu şekilde bir kuyruk nesnesi oluşturulması, kendi MQQueue alt sınıflarınızı yazmanızı sağlar.

Konular

Benzer şekilde, MQQueueManager sınıfının accessTopic yöntemini kullanarak bir konuyu açabilirsiniz. Örneğin, queueManageradlı bir kuyruk yöneticisininde, bir abone ve yayınlayıcı yaratmak için aşağıdaki kodu kullanın:

```
MQTopic subscriber =
      queueManager.accessTopic("TOPICSTRING","TOPICNAME",
      CMQC.MQTOPIC_OPEN_AS_SUBSCRIPTION, CMQC.MQSO_CREATE);
```

```
MQTopic publisher =
      queueManager.accessTopic("TOPICSTRING","TOPICNAME",
      CMQC.MQTOPIC_OPEN_AS_PUBLICATION, CMQC.MQOO_OUTPUT);
```

Konuyu kullanmayı bitirdiğinizde, kapatmak için close () yöntemini kullanın.

Ayrıca, MQTopic oluşturucusunu kullanarak bir konu da yaratabilirsiniz. Parametreler, bir kuyruk yöneticisi değiştirgesinin eklenmesiyle tam olarak accessTopic yöntemiyle aynıdır. Örneğin:

```
MQTopic subscriber = new
      MQTopic(queueManager,"TOPICSTRING","TOPICNAME",
      CMQC.MQTOPIC_OPEN_AS_SUBSCRIPTION, CMQC.MQSO_CREATE);
```

Konular oluştururken bir dizi seçenek belirtebilirsiniz. Bunlara ilişkin ayrıntılar için Class com.ibm.mq.MQTopic başlıklı konuya bakın. Bu şekilde bir konu nesnesi oluşturmak, kendi MQkonusuyla ilgili alt sınıflarınızı yazmanızı sağlar.

Bir konunun yayınlanması için ya da abonelik için açılması gerekir. MQQueueManager sınıfının sekiz accessTopic yöntemi vardır ve Konu sınıfı sekiz oluşturucuda bulunur. Her durumda, bunlardan dördünün bir **destination** parametresi ve dört tanesi de **subscriptionName** parametresine sahiptir (ikisi de dahil olmak üzere). Bunlar yalnızca abonelikler için konuyu açmak için kullanılabilir. Kalan iki yöntemde

bir **openAs** parametresi vardır ve konu, **openAs** parametresinin değerine bağlı olarak yayınlanmak ya da abonelik için açılabilir.

Kalıcı bir abone olarak bir konu oluşturmak için, MQQueueManager sınıfının bir accessTopic yöntemini ya da abonelik adını kabul eden bir MQTopic oluşturucusu kullanın ve her iki durumda da CMQC.MQSO_DURABLE seçeneğini ayarlayın.

Süreçler

Bir sürece erişmek için, MQQueueManager' ın accessProcess yöntemini kullanın. Örneğin, queueManageradlı bir kuyruk yöneticisininde, bir MQProcess nesnesi yaratmak için aşağıdaki kodu kullanın:

```
MQProcess process =
      queueManager.accessProcess("PROCESSNAME",
      CMQC.MQOO_FAIL_IF_QUIESCING);
```

Bir sürece erişmek için, MQQueueManager' ın accessProcess yöntemini kullanın.

accessProcess yöntemi, MQProcess sınıfı yeni bir nesne döndürür.

İşlem nesnesini kullanmayı bitirdiğinizde, aşağıdaki örnekteki gibi kapatmak için close () yöntemini kullanın:

```
process.close();
```

Ayrıca, MQProcess oluşturucusunu kullanarak bir süreç de yaratabilirsiniz. Parametreler, bir kuyruk yöneticisi değiştirgesinin eklenmesiyle tam olarak accessProcess yöntemiyle aynıdır. Örneğin:

```
 MQProcess process =
      new MQProcess(queueManager,"PROCESSNAME",
      CMQC.MQOO_FAIL_IF_QUIESCING);
```

Bu şekilde bir süreç nesnesi oluşturulması, kendi MQProcess alt sınıflarınızı yazmanızı sağlar.

Java için WebSphere MQ sınıflarında iletilerin işlenmesi

İletiler, MQMessage sınıfı tarafından temsil edilir. İletiler, MQQueue ve MQTopic alt sınıfları olan MQDestination sınıfı yöntemlerini kullanarak yerleştirip alır.

MQDestination sınıfının put () yöntemini kullanarak iletileri kuyruklara ya da konulara yerleştirin. MQDestination sınıfının get () yöntemini kullanarak, kuyruklardan ya da konulardan ileti alır. Yordamsal arabirimden farklı olarak, burada MQPUT ve MQGET byte dizileri, Java programlama dili MQMessage sınıfının somut örneklerini alır ve alır. MQMessage sınıfı, gerçek ileti verilerini içeren veri arabelleğini, tüm MQMD (ileti tanımlayıcı) parametrelerini ve bu iletiyi açıklayan ileti özelliklerini içerir.

Yeni bir ileti oluşturmak için, MQMessage sınıfının yeni bir örneğini oluşturun ve ileti arabelleğindeki verileri yerleştirmek için writeXXX yöntemlerini kullanın.

Yeni ileti eşgörünümü yaratıldığında, tüm MQMD parametreleri otomatik olarak varsayılan değerlerine ayarlanır (MQMD için ilk değerler ve dil bildirimleri ' da tanımlandığı gibi). MQDestination 'ın put () yöntemi, değiştirge olarak MQPutMessageSeçenekleri sınıfının bir örneğini de alır. Bu sınıf MQPMO yapısını temsil eder. Aşağıdaki örnek bir ileti yaratır ve bunu bir kuyruğa koyar:

```
// Build a new message containing my age followed by my name
MQMessage myMessage = new MQMessage();
myMessage.writeInt(25);

String name = "Charlie Jordan";
myMessage.writeInt(name.length());
myMessage.writeBytes(name);

// Use the default put message options...
MQPutMessageOptions pmo = new MQPutMessageOptions();
```

```
// put the message!
queue.put(myMessage,pmo);
```

MQDestination 'ın get () yöntemi, kuyruktan yeni alınan iletiyi gösteren yeni bir MQMessage örneği döndürür. Ayrıca, parametre olarak MQGetMessageOptions sınıfının bir eşgörünümünü alır. Bu sınıf MQGMO yapısını temsil eder.

get () yöntemi, gelen iletiye sığması için iç arabelleğindeki büyüklüğü otomatik olarak ayarlandığından, ileti boyutu üst sınırı belirtmeniz gerekmez. Döndürülen iletteki verilere erişmek için MQMessage sınıfının readXXX yöntemlerini kullanın.

Aşağıdaki örnekte, kuyruktan iletinin nasıl alacalacağı gösterilmektedir:

```
// Get a message from the queue
MQMessage theMessage    = new MQMessage();
MQGetMessageOptions gmo = new MQGetMessageOptions();
queue.get(theMessage,gmo);  // has default values

// Extract the message data
int age = theMessage.readInt();
int strLen = theMessage.readInt();
byte[] strData = new byte[strLen];
theMessage.readFully(strData,0,strLen);
String name = new String(strData,0);
```

kodlama üye değişkenini ayarlayarak, okuma ve yazma yöntemlerinin kullandığı sayı biçimini değiştirebilirsiniz.

characterSet adlı üye değişkenini ayarlayarak dizgileri okumak ve yazmak için kullanılacak karakter kümesini değiştirebilirsiniz.

Ek bilgi için “MQMessage sınıfı” sayfa 1028 başlıklı konuya bakın.

Not: MQMessage 'ın writeUTF() yöntemi, içerdiği Unicode baytların yanı sıra dizginin uzunluğunu da otomatik olarak kodlar. İletiniz başka bir Java programı tarafından okunacağı zaman (readUTF() kullanılarak), dizilim bilgileri göndermenin en kolay yoludur.

Java için WebSphere MQ sınıflarında kalıcı olmayan iletilerin performansının geliştirilmesi

İletilere göz atılırken ya da bir istemci uygulamasından kalıcı olmayan iletileri tüketirken performansı artırmak için *devamını oku*seçeneğini kullanabilirsiniz. MQGET ya da zamanuyumsuz tüketimi kullanan istemci uygulamaları, iletiler göz atılırken ya da kalıcı olmayan iletiler tüketirken performans iyileştirmelerinden yararlanacaktır.

İleriye okuma tesisine ilişkin genel bilgiler için bkz. İlgili konu.

Java için WebSphere MQ sınıflarında, CMQC.MQSO_READ_AHEAD ve CMQC.MQSO_NO_READ_AHEAD (MQQueue ya da MQTopic nesnesi), ileti tüketicilerinin ve kuyruk tarayıcılarının o nesne üzerinde okuma yazma izni olup olmadığını saptamak için kullanılır.

Java için WebSphere MQ sınıflarını zamanuyumsuz olarak kullanarak ileti koyma

Bir iletiyi zamanuyumsuz olarak koymak için MQPMO_ASYNC_RESPONSE ayarlayın.

MQDestination sınıfının put () yöntemini kullanarak iletileri kuyruklara ya da konulara yerleştirdiniz. Bir iletiyi zamanuyumsuz olarak yerleştirmek için, kuyruk yöneticisinden yanıt beklemeden işlemin tamamlanmasına izin vermek için, MQPutMessageSeçenekleri 'nin seçenekler alanında MQPMO_ASYNC_RESPONSE ayarlayabilirsiniz. Zamanuyumsuz yerleştirmenin başarısını ya da başarısızlığı saptamak için MQQueueManager.getAsyncDurum çağrısını kullanın.

Java için WebSphere MQ sınıflarında yayınlama/abone olma

Java için WebSphere MQ sınıflarında, konu MQTopic sınıfı tarafından gösterilir ve MQTopic.put() yöntemlerini kullanarak bu konuyu yayınlayabilirsiniz.

WebSphere MQ yayınlama/abone olma hakkında genel bilgi edinmek için bkz. WebSphere MQ yayınlama/abone olma mesajlarına giriş .

Java için WebSphere MQ sınıflarıyla WebSphere MQ ileti üstbilgileri ele alma

Java sınıfları farklı tipte ileti üstbilgisi gösteririr. İki yardımcı sınıfı da sağlanır.

Üstbilgi nesnelerine, üstbilgi alanlarına erişmek ve ileti içeriğini okumak ve yazmak için genel amaçlı yöntemler sağlayan MQHeader arabirimi tarafından açıklanmıştır. Her üstbilgi tipinin, MQHeader arabirimini gerçekleştiren kendi sınıfı vardır ve tek tek alanlar için alıcı ve ayarlayıcı yöntemleri ekler. Örneğin, MQRFH2 üstbilgi tipi MQRFH2 sınıfı tarafından gösterilir; MQDLH sınıfı tarafından MQDLH üstbilgisi ve benzeri bir üstbilgi tipi vardır. Üstbilgi sınıfları, gereken tüm veri dönüştürmeyi otomatik olarak gerçekleştirir ve belirtilen sayısal kodlama ya da karakter kümelerinde (CCSID) veri okuyabilir ya da veri yazabilir.

İki yardımcı sınıf, MQHeaderIterator ve MQHeaderList, iletilerde üstbilgi içeriğini okuma ve çözme (ayrıştırma) yardımcı programı:

- MQHeaderIterator sınıfı, java.util.Iteratorgibi çalışır. İletide daha fazla üstbilgi olduğu sürece, sonraki () yöntemi true değerini döndürür ve nextHeader() ya da next () yöntemi sonraki üstbilgi nesnesini döndürür.
- MQHeaderList , java.util.Listgibi çalışır. MQHeaderIteratorgibi, üstbilgi içeriğini ayrıştırır, ancak aynı zamanda belirli üstbilgileri aramanıza, yeni üstbilgiler eklemenize, var olan üstbilgileri kaldırmanıza, üstbilgi alanlarını güncellemenize ve daha sonra, üstbilgi içeriğini bir iletiye geri yazmanıza olanak tanır. Diğer bir seçenek olarak, boş bir MQHeaderListyaratabilir ve bunu üstbilgi eşgörünümleriyle doldurup bir ya da sürekli olarak bir iletiye yazabilirsiniz.

MQHeaderIterator ve MQHeaderList sınıfları, belirli ileti tipleri ve biçimleriyle ilişkilendirilecek WebSphere MQ üstbilgi sınıflarını öğrenmek için MQHeaderRegistry içindeki bilgileri kullanır. MQHeaderRegistry , tüm geçerli WebSphere MQ biçimleri ve üstbilgi tipleri ve bunların uygulama sınıflarıyla ilgili bilgi içerir ve kendi üstbilgi tiplerinizi kaydettirebilirsiniz.

Yaygın olarak kullanılan Websphere MQ üstbilgileri için destek sağlanır.

- MQRFH-Kurallar ve biçimleme üstbilgisi
- MQRFH2 -Benzer MQRFH gibi, WebSphere Message Broker 'a ait olan bir Message Broker 'a iletileri iletmek için kullanılır. İleti özelliklerini içermek için de kullanılır
- MQCIH- CICS Köprüsü
- MQDLH-Ölü harf üstbilgisi
- MQIIH- IMS bilgi üstbilgisi
- MQRMH-başvuru iletisi üstbilgisi
- MQSAPH- SAP üstbilgisi
- MQWIH-İş bilgisi üstbilgisi
- MQXQH-İletim Kuyruğu üstbilgisi
- MQDH-Dağıtım üstbilgisi
- MQEPH-Kapsüllenmiş PCF üstbilgisi

Ayrıca, kendi üstbilgilerinizin gösterildiği sınıfları da tanımlayabilirsiniz.

RFH2 üstbilgisini almak için MQHeaderIterator kullanmak için, GetMessageSeçenekleri 'nde MQGMO_PROPERTIES_FORCE_MQRFH2 seçeneğini ayarlayın ya da PROPCTL kuyruk özelliğini FORCE olarak ayarlayın.

Java için WebSphere MQ sınıflarını kullanarak bir iletide tüm üstbilgileri yazdırma

Bu örnekte, MQHeaderIterator yönetim ortamı, üstbilgileri bir kuyruktan alınan bir MQMessage 'a ayrıştırır. The MQHeader objects returned from the nextHeader() method display their structure and contents when their toString method is invoked.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQHeader;
import com.ibm.mq.headers.MQHeaderIterator;
...
MQMessage message = ... // Message received from a queue.
MQHeaderIterator it = new MQHeaderIterator (message);

while (it.hasNext ())
{
    MQHeader header = it.nextHeader ();

    System.out.println ("Header type " + header.type () + ": " + header);
}
```

Java için WebSphere MQ sınıflarını kullanarak bir iletide üstbilgiler atlanıyor

Bu örnekte, MQHeaderIterator ' ın skipHeaders() yöntemi, son üstbilginden hemen sonra ileti okuma imlecini konumlandırır.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQHeaderIterator;
...
MQMessage message = ... // Message received from a queue.
MQHeaderIterator it = new MQHeaderIterator (message);

it.skipHeaders ();
```

Java için WebSphere MQ sınıflarını kullanarak gerçek harf iletisinde neden kodunun bulunması

Bu örnekte, okuma yöntemi, iletiden okuyarak MQDLH nesnesini doldurur. Okuma işleminden sonra, ileti okuma imleci, MQDLH üstbilgi içeriğinden hemen sonra konumlanır.

Kuyruk yöneticisinin ölü harf kuyruğunda, önekli iletiler (MQDLH) öneki vardır. Bu iletilerin nasıl işleneceğine karar vermek için-örneğin, bunları yeniden denemek ya da atmak için-bir ölü-mektup işleme uygulamasının MQDLH içinde bulunan neden koduna bakması gerekir.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQDLH;
...
MQMessage message = ... // Message received from the dead-letter queue.
MQDLH dlh = new MQDLH ();

dlh.read (message);

System.out.println ("Reason: " + dlh.getReason ());
```

Tüm üstbilgi sınıfları, doğrudan tek bir adımda doğrudan iletiden kendilerini kullanıma hazırlamak için uygun bir oluşturucu da sağlar. Bu örnekteki kod aşağıdaki gibi basitleştirilebilir:

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQDLH;
...
MQMessage message = ... // Message received from the dead-letter queue.
MQDLH dlh = new MQDLH (message);

System.out.println ("Reason: " + dlh.getReason ());
```

Java için WebSphere MQ sınıflarını kullanarak, MQDLH ' yi okuma yazma ve kaldırma iletisi iletisinden kaldırma

Bu örnekte, üstbilgiyi bir ölü-mektup iletisinden kaldırmak için MQDLH kullanılır.

Bir ölü-mektup işleme uygulaması genellikle, neden kodlarının geçici bir hata olduğunu belirtmesi durumunda reddedilen iletileri yeniden gönderecektir. İletiyi yeniden göndermeden önce, MQDLH üstbilgisini kaldırmalıdır.

Bu örnek, aşağıdaki adımları gerçekleştirir (örneğin, örnek koddaki açıklamalara bakın):

1. MQHeaderList , tüm iletiyi okur ve iletiyle karşılaşılan her üstbilgi, listedeki bir öğe olur.
2. Ölü-harfli iletiler, ilk üstbilgisi olarak bir MQDLH içerir; bu nedenle, bu ileti üstbilgi listesinin ilk öğesinde bulunabilir. MQHeaderList yapılırsa, MQDLH zaten iletiden doldurulmuştur; bu nedenle okuma yöntemini çağılamaya gerek yoktur.
3. Neden kodu, MQDLH sınıfı tarafından sağlanan getReason() yöntemi kullanılarak ayıklanır.
4. Neden kodu incelendi ve iletiyi yeniden göndermenin uygun olduğunu gösterir. MQDLH, MQHeaderList remove () yöntemi kullanılarak kaldırılır.
5. MQHeaderList , kalan içeriğini yeni bir ileti nesnesine yazar. Yeni ileti, MQDLH dışında özgün iletide her şeyi içeriyor ve bir kuyruğa yazılabiliyor. Oluşturucuya ve yazma yöntemine **true** bağımsız değişkeni, ileti gövdesinin MQHeaderListiçinde tutulacağı ve yeniden dışarı yazıldığını gösterir.
6. Şimdi yeni iletinin ileti tanımlayıcısındaki biçim alanı, MQDLH biçim alanında daha önce bulunan değeri içeriyor. İleti verileri, ileti tanımlayıcısındaki sayısal kodlama ve CCSID ayarlarıyla eşleşir.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQDLH;
import com.ibm.mq.headers.MQHeaderList;
...
MQMessage message = ... // Message received from the dead-letter queue.
MQHeaderList list = new MQHeaderList (message, true); // Step 1.
MQDLH dlh = (MQDLH) list.get (0); // Step 2.
int reason = dlh.getReason (); // Step 3.
...
list.remove (dlh); // Step 4.

MQMessage newMessage = new MQMessage ();

list.write (newMessage, true); // Step 5.
newMessage.format = list.getFormat (); // Step 6.
```

Java için WebSphere MQ sınıfları kullanılarak iletinin içeriğinin yazdırılması

Bu örnekte, üstbilgileri de dahil olmak üzere bir iletinin içeriğini yazdırmak için MQHeaderList kullanılır.

Çıktı, iletinin gövdesinin yanı sıra tüm üstbilgi içeriklerinin bir görünümünü de içerir. MQHeaderList sınıfı, bir kerede tüm üstbilgileri kodu çözer; MQHeaderIterator sınıfı, uygulama denetimi altında her defasında birheadersadımına yol gösterirken de bunları çözer. Websphere MQ uygulamalarını yazarken basit bir hata ayıklama aracı sağlamak için bu tekniği kullanabilirsiniz.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQHeaderList;
...
MQMessage message = ... // Message received from a queue.

System.out.println (new MQHeaderList (message, true));
```

Bu örnekte, MQMD sınıfı kullanılarak ileti tanımlayıcı alanları da yazdırılıyor. com.ibm.mq.headers.MQMD sınıfının copyFrom() yöntemi, ileti gövdesinin okunmasından ziyade, MQMessage 'ın ileti tanımlayıcı alanlarından üstbilgi nesnesini doldurur.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQMD;
import com.ibm.mq.headers.MQHeaderList;
...
MQMessage message = ...
MQMD md = new MQMD ();
...
md.copyFrom (message);
System.out.println (md + "\n" + new MQHeaderList (message, true));
```

Java için WebSphere MQ sınıflarını kullanarak bir iletide belirli bir üstbilgi tipini bulma

Bu örnek, varsa bir iletide MQRFH2 üstbilgisi bulmak için MQHeaderList ' un indexOf(String) yöntemini kullanmaktadır.

```
import com.ibm.mq.MQMessage;
import com.ibm.mq.headers.MQHeaderList;
import com.ibm.mq.headers.MQRFH2;
...
MQMessage message = ...
MQHeaderList list = new MQHeaderList (message);
int index = list.indexOf ("MQRFH2");

if (index >= 0)
{
    MQRFH2 rfh = (MQRFH2) list.get (index);
    ...
}
```

Java için WebSphere MQ sınıflarını kullanarak bir MQRFH2 üstbilgisinin çözümleniyor

Bu örnek, MQRFH2 sınıfını kullanarak, adlandırılan bir klasörde bilinen bir alan değerine nasıl erişileceğini gösterir.

MQRFH2 sınıfı, yalnızca yapının sabit bölümündeki alanları değil, aynı zamanda NameValueVeri alanı içinde taşınan XML ile kodlanmış klasör içeriklerine de erişmek için bir dizi yol sağlar. Bu örnek, bir MQ JMS iletisinde yanıt kuyruğu adını temsil eden, jms klasöründeki Rto alanında bilinen bir alan değerine nasıl erişileceğini gösterir.

```
MQRFH2 rfh = ...

String value = rfh.getStringFieldValue ("jms", "Rto");
```

Bir MQRFH2 dosyasının içeriğini keşfetmek için (belirli alanları doğrudan istemek yerine), getFolders listesini, alanlar ve diğer klasörler içerebilecek bir klasörün yapısını temsil eden MQRFH2.Elementlistesini döndürebilirsiniz. Bir alanın ya da klasörün boş değere ayarlanması, alanı MQRFH2' den kaldırır. NameValueVeri klasörü içeriğini bu şekilde değiştirdiğinizde, StrucLength alanı buna göre otomatik olarak güncellenir.

Java için WebSphere MQ sınıflarını kullanarak MQMessage nesnelerine göre byte akışları okuma ve yazma

Bu örnekler, veri kaynağı bir MQMessage nesnesi değilse, WebSphere MQ üstbilgi içeriğini ayrıştırmak ve değiştirmek için üstbilgi sınıflarını kullanır.

You can use the header classes to parse and manipulate WebSphere MQ header content even when the data source is something other than an MQMessage object. Her üstbilgi sınıfı tarafından uygulanan MQHeader arabirimi, `int read (java.io.DataInput message, int encoding, int characterSet)` ve `int write (java.io.DataOutput message, int encoding, int characterSet)` yöntemlerini sağlar. com.ibm.mq.MQMessage sınıfı, java.io.DataInput ve java.io.DataOutput arabirimlerini uygular. Bu, ileti tanımlayıcısında belirtilen kodlamayı ve CCSID ' yi geçersiz kılarak, MQMessage içeriğini okumak ve yazmak için iki MQHeader yöntemini kullanabilmeniz anlamına gelir. Bu, farklı kodlamalarda üstbilgi zincirini içeren iletiler için kullanışlıdır.

Ayrıca, diğer veri akışlarından DataInput ve DataOutput nesnelerini, örneğin dosya ya da yuva akımları ya da JMS iletilerinde taşınan bayt dizileri elde edebilirsiniz. The java.io.DataInputStream classes implement DataInput and the java.io.DataOutputStream classes implement DataOutput. Bu örnek, bir bayt dizisinden WebSphere MQ üstbilgi içeriğini okur:

```
import java.io.*;
import com.ibm.mq.headers.*;
...
byte [] bytes = ...
DataInput in = new DataInputStream (new ByteArrayInputStream (bytes));
MQHeaderIterator it = new MQHeaderIterator (in, CMQC.MQENC_NATIVE,
  CMQC.MQCCSI_DEFAULT);
```

The line starting MQHeaderIterator could be replaced with

```
MQDLH dlh = new MQDLH (in, CMQC.MQENC_NATIVE, CMQC.MQCCSI_DEFAULT);
// or any other header type
```

Bu örnek, DataOutputAkımı kullanılarak bir bayt dizisine yazar:

```
MQHeader header = ... // Could be any header type
ByteArrayOutputStream out = new ByteArrayOutputStream ();

header.write (new DataOutputStream (out), CMQC.MQENC_NATIVE, CMQC.MQCCSI_DEFAULT);
byte [] bytes = out.toByteArray ();
```

Bu şekilde akışlarla çalıştığınızda, kodlama ve characterSet bağımsız değişkenleri için doğru değerleri kullanmaya dikkat edin. Üstbilgileri okurken, byte içeriğinin başlangıçta yazıldığı kodlamayı ve CCSID ' yi belirtin. Üstbilgi yazarken, oluşturmak istediğiniz kodlamayı ve CCSID ' yi belirtin. Veri dönüştürme otomatik olarak üstbilgi sınıflarına göre gerçekleştirilir.

Java için WebSphere MQ sınıflarını kullanarak yeni üstbilgi tipleri için sınıflar yaratılması

Java için WebSphere MQ sınıflarıyla sağlanmamamış üstbilgi tipleri için Java sınıfları yaratabilirsiniz.

Java için WebSphere MQ sınıflarıyla sağlanan herhangi bir üstbilgi sınıfı ile aynı şekilde kullanabileceğiniz yeni bir üstbilgi tipini simgeleyen bir Java sınıfı eklemek için, MQHeader arabirimini gerçekleştiren bir sınıf yaratabilirsiniz. En basit yaklaşım, com.ibm.mq.headers.impl.Header sınıfını genişletmeniz olmalıdır. Bu örnek, MQTM üstbilgi yapısını gösteren tam işlevli bir sınıf üretir. Her alan için ayrı alıcı ve ayarlayıcı (setter) yöntemleri eklemeniz gerekmez; ancak, üstbilgi sınıfı kullanıcıları için yararlı bir kolaylık sağlar. Alan adı için bir dizgi alan soysal getValue ve setValue yöntemleri, üstbilgi tipinde tanımlanan tüm alanlar için çalışır. Edinilmiş okuma, yazma ve büyüklük yöntemleri, yeni üstbilgi tipinin eşgörünümlerinin okunmasını ve yazılacağını ve üstbilgi büyüklüğünü, alan tanımlamasına dayalı olarak doğru olarak hesaplayacaklarını sağlar. Tip tanımlaması yalnızca bir kez yaratılır, ancak bu üstbilgi sınıfının birçok eşgörünümü yaratılır. To make the new header definition available for decoding using the MQHeaderIterator or MQHeaderList classes, you would register it using the MQHeaderRegistry. Ancak, MQTM üstbilgi sınıfının zaten bu pakette yer aldığına ve varsayılan kayıt dosyasına kaydedildiğine dikkat edin.

```
import com.ibm.mq.headers.impl.Header;
import com.ibm.mq.headers.impl.HeaderField;
import com.ibm.mq.headers.CMQC;

public class MQTM extends Header {
    final static HeaderType TYPE = new HeaderType ("MQTM");
    final static HeaderField StrucId = TYPE.addMQChar ("StrucId", CMQC.MQTM_STRUC_ID);
    final static HeaderField Version = TYPE.addMQLong ("Version", CMQC.MQTM_VERSION_1);
    final static HeaderField QName = TYPE.addMQChar ("QName", CMQC.MQ_Q_NAME_LENGTH);
    final static HeaderField ProcessName = TYPE.addMQChar ("ProcessName",
      CMQC.MQ_PROCESS_NAME_LENGTH);
    final static HeaderField TriggerData = TYPE.addMQChar ("TriggerData",
      CMQC.MQ_TRIGGER_DATA_LENGTH);
    final static HeaderField ApplType = TYPE.addMQLong ("ApplType");
    final static HeaderField ApplId = TYPE.addMQChar ("ApplId", 256);
    final static HeaderField EnvData = TYPE.addMQChar ("EnvData", 128);
    final static HeaderField UserData = TYPE.addMQChar ("UserData", 128);

    protected MQTM (HeaderType type){
        super (type);
    }
    public String getStrucId () {
        return getStringValue (StrucId);
    }
    public int getVersion () {
        return getIntValue (Version);
    }
    public String getQName () {
        return getStringValue (QName);
    }
    public void setQName (String value) {
        setStringValue (QName, value);
    }
    // ...Add convenience getters and setters for remaining fields in the same way.
}
```

Java için WebSphere MQ sınıflarıyla PCF iletilerinin işlenmesi

Java sınıfları PCF tarafından yapılandırılmış iletiler yaratmak ve ayrıştırmak ve PCF isteklerini göndermek ve PCF yanıtlarını toplamak için sağlanmıştır.

PCFMessage & MQCFGR sınıfları, PCF parametre yapılarının dizilerini temsil eder. Bunlar, PCF parametreleri eklemek ve almak için kolaylık sağlayan yöntemler sağlar.

PCF değiştirge yapıları, MQCFH, MQCFIN, MQCFIN64, MQCFST, MQCFBS, MQCFIL, MQCFIL64 MQCFSL ve MQCFGR sınıflarıyla gösterilir. Bu paylaşıma ilişkin temel işletim arabirimleri şunlardır:

- İleti içeriğini okuma ve yazma yöntemleri: read (), write () ve boyut ()
- Değiştirgelerin işlenmesine ilişkin yöntemler: getValue (), setValue (), getParameter () ve diğerleri
- Enumerator yöntemi.nextParameter (), bir MQMessage 'da PCF içeriğini ayrıştırıyor

Bir süzgeç işlevi sağlamak için, sorgu komutlarında PCF süzgeç değiştirgesi kullanılır. Aşağıdaki sınıflarda kapsüllenmiş:

- MQCFIF-tamsayı süzgeci
- MQCFSF-dizgi süzgeci
- MQCFBF-byte süzgeci

İki aracı sınıfı, PCFAgent ve PCFMessageAgent , bir Kuyruk Yöneticisine, komut sunucusu kuyruğuna ve ilişkili bir yanıt kuyruğuna bağlantıyı yönetmek için sağlanır. PCFMessageAgent , PCFAgent 'ı genişletir ve olağan durumda tercihte kullanılır. PCFMessageAgent sınıfı, alınan MQMessages öğesini dönüştürür ve bunları, bir PCFMessage dizisi olarak çağırıcıya geri çevirir. PCFAgent, kullanmadan önce ayrıştırmak zorunda olduğunuz bir MQMessages dizisini döndürür.

Java için WebSphere MQ sınıflarında ileti özelliklerinin işlenmesi

İşlem ileti tanıtıcılarına yönelik işlev çağrılarının, Java için WebSphere MQ sınıflarında eşdeğer bir değeri yoktur. İleti tanıtıcısı özelliklerini ayarlamak, geri döndürmek ya da silmek için, MQMessage sınıfının yöntemlerini kullanın.

İleti özelliklerine ilişkin genel bilgi için bkz. “Özellik adları” sayfa 18.

İletilere Java erişimi için WebSphere MQ sınıflarında MQMessage sınıfı geçmektedir. Bu nedenle, ileti tanıtıcıları Java ortamında sağlanmamaktadır ve WebSphere MQ işlevi, MQCRTMH, MQDLTMH, MQMHBUF ve MQBUFMH çağrılarına eşdeğer değildir.

Yordamsal arabirimde ileti tanıtıcısı özelliklerini ayarlamak için, MQSETMP çağrısını kullanıyorsunuz. Java için WebSphere MQ sınıflarında, MQMessage sınıfı için uygun yöntemi kullanın:

- setBooleanözelliği
- setByteözelliği
- setBytesÖzelliği
- setShortözelliği
- setIntözelliği
- setInt2Property
- setInt4Property
- setInt8Property
- setLongözelliği
- setFloatÖzelliği
- setDoubleözelliği
- setStringözelliği
- setObjectözelliği

Bunlar bazen toplu olarak *set*property* yöntemleri olarak adlandırılır.

Yordamsal arabirimde ileti tanıtıcısı özelliklerinin değerini döndürmek için, MQINQMP çağrısını kullanıyorsunuz. Java için WebSphere MQ sınıflarında, MQMessage sınıfı için uygun yöntemi kullanın:

- getBooleanÖzelliği
- getByteÖzelliği
- getBytesÖzelliği
- getShortÖzelliği
- getIntÖzelliği
- getInt2Property
- getInt4Property
- getInt8Property
- getLongÖzelliği
- getFloatÖzelliği
- getDoubleÖzelliği
- getStringÖzelliği
- getObjectÖzelliği

Bunlar bazen toplu olarak *get*property* yöntemleri olarak adlandırılır.

Yordamsal arabirimde ileti tanıtıcısı özelliklerinin değerini silmek için, MQDLTMP çağrısını kullanıyorsunuz. Java için WebSphere MQ sınıflarında, MQMessage sınıfının deleteProperty yöntemini kullanın.

Java için WebSphere MQ sınıflarında hataların işlenmesi

Java `try` ve `catch` öbekleri kullanılarak Java için WebSphere MQ sınıflarından kaynaklanan hatalarla başa çıkabilr.

Java arabirimindeki yöntemler bir tamamlanma kodu ve neden kodu döndürmez. Bunun yerine, bir WebSphere MQ çağrısından kaynaklanan tamamlanma kodu ve neden kodu her ikisi de sıfır değilse, bir kural dışı durum yayınlarlar. Bu, her bir WebSphere MQçağrısından sonra dönüş kodlarını kontrol etmek zorunda kalmamak için program mantığını basitleştirir. Programınızın hangi noktalarda hata olma olasılığına karşı karar vereceğine karar verebilirsiniz. Bu noktalarda kodunuzu `try` ve `catch` blokları ile çevrebilirsiniz. Örneğin:

```
try {
    myQueue.put(messageA,putMessageOptionsA);
    myQueue.put(messageB,putMessageOptionsB);
}
catch (MQException ex) {
    // This block of code is only executed if one of
    // the two put methods gave rise to a non-zero
    // completion code or reason code.
    System.out.println("An error occurred during the put operation:" +
                           "CC = " + ex.completionCode +
                           "RC = " + ex.reasonCode);
    System.out.println("Cause exception:" + ex.getCause() );
}
```

z/OS için Java özel durumlarında bildirilen WebSphere MQ arama neden kodları, diğer tüm altyapılar için z/OSiçin neden kodları ve Neden kodları ' te belgelenir.

Java uygulaması için bir WebSphere MQ sınıfları çalıştırılırken yayınlanan kural dışı durumlar da günlüğe yazılır. Ancak bir uygulama, belirli bir neden koduyla ilişkili özel durumları günlüğe kaydedilmesini önlemek için MQException.logExclude() yöntemini çağırabilir. Belirli bir neden koduyla ilişkili birçok kural dışı durumun ortaya atılacağı ve günlüğünün bu kural dışı durumlarla doldurulmamasını beklediğiniz durumlarda bunu yapmak isteyebilirsiniz. Örneğin, uygulamanız bir döngü etrafında her yinelendiğinde bir kuyruktan ileti almayı denerse ve bu girişimlerin çoğu için, kuyrukta uygun bir ileti olmadığını bekliyorsanız, MQRC_NO_MSG_AVAILEABLE ' nin günlüğe kaydedilmesinin neden koduyla ilişkili kural dışı durumları önlemek isteyebilirsiniz. If an application has previously prevented exceptions associated

with a specific reason code from being logged, it can allow these exceptions to be logged again by calling the method MQException.logInclude().

Bazen neden kodu, hatayla ilişkili tüm ayrıntıları iletmez. Yayınlanan her kural dışı durum için, bir uygulamanın bağlantılı kural dışı durumu denetmesi gerekir. Bağlantılı kural dışı durumun kendisi başka bir bağlantılı kural dışı duruma sahip olabilir ve bu nedenle bağlantılı kural dışı durumlar, özgün temel soruna geri giden bir zincir oluşturur. java.lang.Throwable sınıfının zincirleme kural dışı durum mekanizması kullanılarak bağlantılı bir kural dışı durum uygulandı ve bir uygulama, Throwable.getCause() yöntemini çağırarak bağlantılı bir kural dışı durumu alır. Bir MQException örneği olan bir kural dışı durumdan MQException.getCause(), temeldeki com.ibm.mq.jmqi.JmqiExceptionve getCause örneğini alır; bu kural dışı durum, hataya neden olan temel java.lang.Exception değerini alır.

MQException sınıfı varsayılan olarak, genellikle konsola yönlendirilen System.errkural dışı durumlarını otomatik olarak akıtır. Konsolda yer alan kural dışı durumları durdurmak istiyorsanız, uygulamanıza MQException.log= null değerini ayarlamak için bir satır ekleyin.

Java için WebSphere MQ sınıflarında öznitelik değerlerini alma ve ayarlama

Birçok ortak öznitelik içingetXXX() ve setXXX() yöntemleri sağlanmıştır. Diğerlerine soysal sorma () ve set () yöntemleri kullanılarak erişilebilir.

For many of the common attributes, the classes MQManagedObject, MQDestination, MQQueue, MQTopic, MQProcess, and MQQueueManager contain getXXX() and setXXX() methods. Bu yöntemler, öznitelik değerlerini almanıza ve ayarlamanıza olanak sağlar. MQDestination, MQQueue ve MQTopic için, yöntemlerin yalnızca, nesneyi açtığınızda uygun sorgu ve küme işaretlerini belirttiğinizde işe yaradığını unutmayın.

Daha az ortak öznitelikler için, MQQueueManager, MQDestination, MQQueue, MQTopic, ve MQProcess sınıfları tüm devralırları MQManagedObjectadlı bir sınıftan devralır. Bu sınıf, sorgula () ve set () arabirimlerini tanımlar.

Yeni işlecini kullanarak yeni bir kuyruk yöneticisi nesnesi yarattığınızda, bu nesne otomatik olarak sorgulanmak üzere açılır. Bir süreç nesnesine erişmek için accessProcess() yöntemini kullandığınızda, o nesne sorgulamak için otomatik olarak açılır. Bir kuyruk nesnesine erişmek için accessQueue() yöntemini kullandığınızda, bu nesne sorgulama ya da ayarlama işlemleri için *değil* otomatik olarak açılır. Bunun nedeni, bu seçeneklerin otomatik olarak eklenmesi bazı uzak kuyruklar tipleriyle sorunlara neden olabilir. Bir kuyrukta sorgula, set, getXXXve setXXX yöntemlerini kullanmak için, accessQueue() yönteminin openOptions değiştirgesinde uygun olarak sorgu ve ayar işaretlerini belirlemeniz gerekir. Aynı durum hedef ve konu nesneleri için de geçerlidir.

Sorgu ve ayarlama yöntemleri üç parametre alır:

- seçiciler dizisi
- intAttrs dizisi
- charAttrs dizisi

Java 'daki bir dizinin uzunluğu her zaman bilindiğinden, MQINQ ' da bulunan SelectorCount, IntAttrCount ve CharAttrLength değiştirgelerine gerek yoktur. Aşağıdaki örnek, bir kuyruğun nasıl bir kuyruğun üzerinde nasıl yapılır gösterileceğini göstermektedir:

```
// inquire on a queue
final static int MQIA_DEF_PRIORITY = 6;
final static int MQCA_Q_DESC = 2013;
final static int MQ_Q_DESC_LENGTH = 64;

int[] selectors  = new int[2];
int[] intAttrs   = new int[1];
byte[] charAttrs = new byte[MQ_Q_DESC_LENGTH]

selectors[0] = MQIA_DEF_PRIORITY;
selectors[1] = MQCA_Q_DESC;

queue.inquire(selectors,intAttrs,charAttrs);
```

```
System.out.println("Default Priority = " + intAttrs[0]);
System.out.println("Description : " + new String(charAttrs,0));
```

Java 'da çok iş parçacıklı programlar

Java çalıştırma ortamı doğal olarak çok iş parçacıklıdır. Java için WebSphere MQ sınıfları, bir kuyruk yöneticisi nesnesinin birden çok iş parçacığı tarafından paylaşılmasını sağlar, ancak hedef kuyruk yöneticisine tüm erişimin uyumlulaştırılmasını sağlar.

Birden çok iş parçacıklı programlar Java 'da saklanmamak için zordur. Bir kuyruk yöneticisine bağlanan ve başlatma sırasında kuyruk açan basit bir programı düşünün. Program ekranda tek bir düğme görüntüler. Bir kullanıcı bu düğmeyi tıklattığında, program kuyruktan bir ileti alır.

Java çalıştırma ortamı doğal olarak çok iş parçacıklıdır. Bu nedenle, uygulamanızın kullanıma hazırlanması bir iş parçacığıda gerçekleşir ve düğmenin düğmesine yanıt olarak yürütülen kod ayrı bir iş parçacığıda (kullanıcı arabirimi iş parçacığı) yürütülür.

C tabanlı WebSphere MQ MQI istemcisi ile, birden çok iş parçacığının çekme noktalarının paylaşılmasına ilişkin sınırlamalar olduğu için bu, bir soruna neden olur. Java için WebSphere MQ sınıfları bu kısıtı aktarıp, bir kuyruk yöneticisi nesnesinin (ve ilişkili kuyruk, konu ve süreç nesnelerinin) birden çok iş parçacığının paylaşılmasına olanak tanır.

Java için WebSphere MQ sınıflarının uygulanması, belirli bir bağlantı (MQQueueManager nesne eşgörünümü) için, hedef WebSphere MQ kuyruk yöneticisine tüm erişimin uyumlulaştırılmasını sağlar. Bir kuyruk yöneticisine çağrı yapmak isteyen bir iş parçacığı, ilgili bağlantı için devam eden diğer tüm çağrılar tamamlanıncaya kadar engellenir. Programınızdaki birden çok iş parçacığının aynı kuyruk yöneticisine eşzamanlı olarak erişmeniz gerekiyorsa, eşzamanlı erişim gerektiren her iş parçacığı için yeni bir MQQueueManager nesnesi yaratın. (Bu, her iş parçacığı için ayrı bir MQCONN çağrısı yayınlamaya eşdeğerdir.)

Not: Aynı anda ileti isteyen iş parçacıkları arasında `com.ibm.mq.MQGetMessageOptions` sınıfı eşgörünümleri paylaşılmamalıdır. Bu sınıfın eşgörünümleri, ilgili MQGET isteği sırasında verilerle güncellenir ve birden çok iş parçacığı nesnenin aynı eşgörünümünde koşutzamanlı olarak çalışırken beklenmeyen sonuçlarla sonuçlanabilir.

Java için WebSphere MQ sınıflarında kanal çıkışlarının kullanılması

Java için WebSphere MQ sınıflarını kullanarak bir uygulamadaki kanal çıkışlarının nasıl kullanılacağı hakkında genel bakış.

Aşağıdaki konularda, bir kanal çıkışısının Java 'da nasıl yazılacağı, nasıl atanacağını ve verilerin nasıl aktarılacağı açıklanmaktadır. Daha sonra, C içinde yazılan kanal çıkışlarının nasıl kullanılacağını ve kanal çıkışlarının sırasını nasıl kullanacaklarını açıklar.

Uygulamanızın kanal çıkış sınıfını yüklemek için doğru güvenlik iznine sahip olması gerekir.

Java için WebSphere MQ sınıflarında kanal çıkışı yaratılması

Uygun bir arabirimi gerçekleştiren bir Java sınıfı tanımlayarak kendi kanal çıkışlarınızı sağlayabilirsiniz.

Bir çıkışı gerçekleştirmek için, uygun arabirimi gerçekleştiren yeni bir Java sınıfı tanımlarsiniz. com.ibm.mq.exits paketindeki üç çıkış arabirimi tanımlanir:

- WMQSendExit
- WMQReceiveExit
- WMQSecurityExit

Not: Kanal çıkışları yalnızca istemci bağlantıları için desteklenir; bağ tanımları bağlantıları için desteklenmez. Java için WebSphere MQ sınıflarının dışında bir Java kanalı çıkışı kullanamazsınız; örneğin, C içinde yazılmış bir istemci uygulaması kullanıyorsanız.

Bir bağlantı için tanımlanan SSL şifrelemesi, *bundan sonra* gönderme ve güvenlik çıkışları çağrılır. Benzer şekilde, şifre çözme işlemi *önce* alma ve güvenlik çıkışları çağrılmadan gerçekleştirilir.

Aşağıdaki örnekte, üç arabirimi gerçekleştiren bir sınıf tanımlanmaktadır:

```
public class MyMQExits implements
WMQSendExit, WMQReceiveExit, WMQSecurityExit {
    // Default constructor
  public MyMQExits(){
  }
    // This method comes from the send exit interface
  public ByteBuffer channelSendExit(
MQCXP channelExitParms,
                                    MQCD channelDefinition,
                                    ByteBuffer agentBuffer)
  {
    // Fill in the body of the send exit here
  }
    // This method comes from the receive exit interface
  public ByteBuffer channelReceiveExit(
MQCXP channelExitParms,
                                      MQCD channelDefinition,
                                      ByteBuffer agentBuffer)
  {
    // Fill in the body of the receive exit here
  }
    // This method comes from the security exit interface
  public ByteBuffer channelSecurityExit(
MQCXP channelExitParms,
                                       MQCD channelDefinition,
                                       ByteBuffer agentBuffer)
  {
    // Fill in the body of the security exit here
  }
}
```

Her çıkışta bir MQCXP nesnesi ve bir MQCD nesnesi geçirilir. Bu nesneler, yordamsal arabirimde tanımlanan MQCXP ve MQCD yapılarını gösterir.

Yazdığınız herhangi bir çıkış sınıfının bir oluşturucusu olmalıdır. Bu, varsayılan oluşturucu ya da bir dizgi bağımsız değişkeni alan bir oluşturucu olabilir. Bir dizgi sürerse, kullanıcı verileri yaratıldığında çıkış sınıfına aktarılır. Çıkış sınıfı hem varsayılan bir oluşturucu, hem de tek bir bağımsız değişken oluşturucusu içeriyorsa, tek bağımsız değişken oluşturucusunun önceliği vardır.

Gönderme ve güvenlik çıkışlarında, çıkış kodunuz sunucuya göndermek istediğiniz verileri döndürmelidir. Bir alma çıkışı için çıkış kodunuz, WebSphere MQ ' un yorumlayacağı değiştirilmiş verileri döndürmelidir.

Olabilecek en basit çıkış gövdesi şunlardır:

```
{ return agentBuffer; }
```

Kuyruk yöneticisini kanal çıkışı içinden kapatmayın.

Varolan kanal çıkış sınıflarının kullanılması

7.0sürümünden önceki WebSphere MQ sürümlerinde, aşağıdaki örnekte olduğu gibi MQSendExit, MQReceiveExitve MQSecurityExitarabirimlerini kullanarak bu çıkışları uygulardınız. Bu yöntem geçerli olmaya devam eder, ancak geliştirilmiş işlev ve başarım için yeni yöntem tercih edilir.

```
public class MyMQExits implements MQSendExit, MQReceiveExit, MQSecurityExit {
    // Default constructor
  public MyMQExits(){
  }
    // This method comes from the send exit
  public byte[] sendExit(MQChannelExit channelExitParms,
                         MQChannelDefinition channelDefParms,
                         byte agentBuffer[])
  {
    // Fill in the body of the send exit here
  }
    // This method comes from the receive exit
  public byte[] receiveExit(MQChannelExit channelExitParms,
                            MQChannelDefinition channelDefParms,
                            byte agentBuffer[])
  {
    // Fill in the body of the receive exit here
```

```
  }
    // This method comes from the security exit
  public byte[] securityExit(MQChannelExit channelExitParms,
                             MQChannelDefinition channelDefParms,
                             byte agentBuffer[])
  {
    // Fill in the body of the security exit here
  }
}
```

Assigning a channel exit in IBM WebSphere MQ classes for Java

IBM WebSphere MQ classes for Javaolanağını kullanarak bir kanal çıkışı atayabilirsiniz.

IBM WebSphere MQ classes for Javaiçindeki IBM WebSphere MQ kanalına doğrudan eşdeğer bir değer yoktur. Kanal çıkışları bir MQQueueManager' a atanılır. Örneğin, WMQSecurityExit arabirimini gerçekleştiren bir sınıfı tanımlamış olmak için, bir uygulama güvenlik çıkışını şu dört yoldan biriyle kullanabilir:

- Bir MQQueueManager nesnesi yaratmadan önce, sınıfın bir eşgörünümünü MQEnvironment.channelSecurityExit alanına atayarak
- By setting the MQEnvironment.channelSecurityExit field to a string representing the security exit class before creating an MQQueueManager object
- By creating a key/value pair in the properties hashtable passed to MQQueueManager with a key of CMQC.SECURITY_EXIT_PROPERTY
- İstemci kanal tanımlama çizelgesi (CCDT) kullanılması

MQEnvironment.channelSecurityExit alanını bir dizgiye ayarlayarak, özellikler hashtable ya da CCDT kullanarak bir anahtar/değer çifti oluşturarak atanacak herhangi bir çıkış, varsayılan bir oluşturucuyla yazılmalıdır. Bir sınıfın somut örneği olarak atanan bir çıkışa, uygulamaya bağlı olarak varsayılan bir oluşturucuya gerek yoktur.

Bir uygulama, benzer şekilde bir gönderme ya da alma çıkışını kullanabilir. Örneğin, aşağıdaki kod parçası, daha önce MQEnvironment kullanarak tanımlanan MyMQExitssınıfında uygulanan güvenlik, gönderme ve alma çıkışlarını nasıl kullanacağını gösterir.

```
MyMQExits myexits = new MyMQExits();
MQEnvironment.channelSecurityExit = myexits;
MQEnvironment.channelSendExit = myexits;
MQEnvironment.channelReceiveExit = myexits;
:
MQQueueManager jupiter = new MQQueueManager("JUPITER");
```

Kanal çıkışı atamak için birden çok yöntem kullanılırsa, öncelik sırası aşağıdaki gibidir:

1. Bir CCDT 'nin URL adresi MQQueueManager' a iletilirse, CCDT ' nin içeriği kullanılacak kanal çıkışlarını ve MQEnvironment 'daki çıkış tanımlarını ya da özellik gruplama çizelgesini (hashtable) yoksayılır.
2. CCDT URL iletilmezse, MQEnvironment 'dan çıkış tanımlamaları ve gruplama çizelgesi birleştirilir
   - Aynı çıkış tipi hem MQEnvironment, hem de hashtable içinde tanımlandıysa, hashtable içinde tanım kullanılır.
   - If equivalent old and new types of exit are specified (for example the sendExit field, which can only be used for the type of exit used in versions of IBM WebSphere MQ earlier than Version 7.0, and the channelSendExit field, which can be used for any send exit), the new exit (channelSendExit) is used rather than the old exit.

Bir kanal çıkışı dizgi olarak bildirdiyseniz, kanal çıkış programının yerini belirlemek için IBM WebSphere MQ ' i etkinleştirmeniz gerekir. Uygulamanın çalışmakta olduğu ortama ve kanal çıkış programlarının nasıl paketleneceği üzerine bağlı olarak çeşitli şekillerde yapabilirsiniz.

- Bir uygulama sunucusunda çalışan bir uygulama için, dosyaları Çizelge 88 sayfa 662 içinde gösterilen ya da **exitClasspath**tarafından başvurulan JAR dosyalarında paketlenmiş bir dizine saklamanız gerekir.
- Uygulama sunucusunda çalışmayan bir uygulama için aşağıdaki kurallar geçerli olur:

– Kanal çıkış sınıflarınız ayrı JAR dosyalarında paketlendiyse, bu JAR dosyaları **exitClasspath**' te yer almalıdır.
– Kanal çıkış sınıflarınız JAR dosyalarında paketlenmediyse, sınıf dosyaları Çizelge 88 sayfa 662 içinde gösterilen dizinde ya da JVM sistem sınıfı yolundaki herhangi bir dizinde ya da **exitClasspath**dizininde saklanabilir.

exitClasspath özelliği dört şekilde belirtilebilir. Öncelik sırasına göre, bu yollar aşağıdaki gibidir:

1. The system property com.ibm.mq.exitClasspath (defined on the command line using the `-D` option)
2. `mqclient.ini` dosyasının exitPath kısmı
3. A hashtable entry with the key CMQC.EXIT_CLASSPATH_PROPERTY
4. MQEnvironment değişkeni **exitClasspath**

Birden çok yolu java.io.File.pathSeparator karakterini kullanarak ayırın.

Çizelge 88. Kanal çıkış programlarına ilişkin dizin

Altyapı	**Dizin**
AIX, HP-UX, Linuxve Solaris	`/var/mqm/exits` (32-bit kanal çıkış programları) `/var/mqm/exits64` (64-bit kanal çıkış programları)
Windows	*kuruluş_data_dzn*\çıkışlar
Not: *install_data_dir* , kuruluş sırasında IBM WebSphere MQ veri dosyaları için seçtiğiniz dizindir. Varsayılan dizin `C:\Program Files\IBM\WebSphere MQ`dizinidir.	

Java için WebSphere MQ sınıflarındaki kanal çıkışlarına veri aktarma

Kanal çıkışlarına veri aktarabilir ve kanal çıkışlarından uygulamanıza veri döndürebilirsiniz.

agentBuffer parametresi

Gönderme çıkışı için, *agentBuffer* parametresi, gönderilecek verileri içerir. Bir alma çıkışı ya da güvenlik çıkışı için, *agentBuffer* parametresi az önce alınmış verileri içerir. agentBuffer.limit () ifadesi dizinin uzunluğunu gösterdiğinden, uzunluk parametresine gerek yoktur.

Gönderme ve güvenlik çıkışlarında, çıkış kodunuz sunucuya göndermek istediğiniz verileri döndürmelidir. Bir alma çıkışı için çıkış kodunuz, WebSphere MQ ' un yorumlayacağı değiştirilmiş verileri döndürmelidir.

Olabilecek en basit çıkış gövdesi şunlardır:

```
{ return agentBuffer; }
```

Kanal çıkışları, arka diziye sahip bir arabelleğle çağrılır. En iyi başarımı elde etmek için, çıkışta yedek diziye sahip bir arabellek döndürülmelidir.

Kullanıcı verileri

If an application connects to a queue manager by setting channelSecurityExit, channelSendExit, or channelReceiveExit, 32 bytes of user data can be passed to the appropriate channel exit class when it is called, using the channelSecurityExitUserData, channelSendExitUserData, or channelReceiveExitUserData fields. Bu kullanıcı verileri kanal çıkış sınıfı için kullanılabilir, ancak çıkışta her çağrıldığında yenilenir. Bu nedenle, kanal çıkışındaki kullanıcı verilerinde yapılan değişiklikler kaybedilir. Bir kanal çıkışındaki verilerde kalıcı değişiklikler yapmak istiyorsanız, MQCXP exitUseralanını kullanın. Çıkışa yönelik çağrılar arasında bu alandaki veriler saklanır.

Uygulama securityExit, sendExitya da receiveExitayarlarsa, bu kanal çıkış sınıflarına kullanıcı verisi aktarılabilir.

Bir uygulama, bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, bir istemci bağlantı kanalı tanımlamasında belirtilen kullanıcı verileri, çağrıldığında kanal çıkış sınıflarına geçirilir. İstemci kanal tanımlama çizelgesini kullanma hakkında daha fazla bilgi için bkz. “Using a client channel definition table with IBM WebSphere MQ classes for Java” sayfa 645.

Java için WebSphere MQ sınıflarıyla Java 'da yazılmamış kanal çıkışlarını kullanma

Bir Java uygulamasından C içinde yazılan kanal çıkış programlarını nasıl kullanacağını

In WebSphere MQ Version 7.0, you can specify the name of a channel exit program written in C as a String passed to the channelSecurityExit, channelSendExit, or channelReceiveExit fields in the MQEnvironment object or properties Hashtable. Ancak, başka bir dilde yazılmış bir uygulamadaki Java dilinde yazılmış bir kanal çıkışı kullanamazsınız.

Specify the exit program name in the format `library(function)` and ensure that the location of the exit program is included in the path environment variable.

C içinde bir kanal çıkışı nasıl yazılabilmeye ilişkin bilgi için bkz. “İleti alışverişi kanallarına ilişkin kanal çıkışı programları” sayfa 378.

Dış çıkış sınıflarının kullanılması

In versions of WebSphere MQ earlier than Version 7.0, three classes were provided to enable you to use channel exits written in languages other than Java:

- MQSecurityExit arabirimini gerçekleştirenMQExternalSecurityExit
- MQSendExit arabirimini gerçekleştirenMQExternalSendExit
- MQReceiveExit arabirimini gerçekleştirenMQExternalReceiveExit

Bu sınıfların kullanımı geçerli olmaya devam eder, ancak yeni yöntem tercih edilir.

Java 'da yazılmamış bir güvenlik çıkışı kullanmak için, önce uygulama bir MQExternalSecurityçıkış nesnesi yaratmış olmalıdır. Belirtilen uygulama, MQExternalSecurityçıkış oluşturucusuna parametre olarak, güvenlik çıkışını içeren kitaplığın adı, güvenlik çıkışa ilişkin giriş noktasının adı ve çağrıldığında güvenlik çıkışa geçirilecek kullanıcı verileri. Java 'da yazılmamış olan kanal çıkış programları, Çizelge 88 sayfa 662içinde gösterilen dizinde depolanır.

Java için WebSphere MQ sınıflarında bir dizi kanal gönderme ya da alma çıkışı kullanma

Java uygulaması için bir WebSphere MQ sınıfları, art arda çalıştırılan bir kanal gönderme ya da alma çıkış dizisi kullanabilir.

Bir uygulama çıkış dizisini kullanmak için, bir uygulama gönderme çıkışlarını içeren bir Liste ya da Dizgi yaratabilir. Bir Liste kullanılırsa, Liste 'nin her öğesi aşağıdakilerden biri olabilir:

- WMQSendExit arabirimini gerçekleştiren kullanıcı tanımlı bir sınıfın somut örneği
- MQSendExit arabirimini gerçekleştiren kullanıcı tanımlı bir sınıfın somut örneği (Java 'da yazılan bir gönderme çıkışı için)
- MQExternalSendçıkış sınıfının somut örneği (Java 'da bir gönderme çıkışı için yazılmamış)
- MQSendExitZincir sınıfının somut örneği
- Dizgi sınıfının somut örneği

Bir Liste başka bir Liste içeremez.

Uygulama, benzer bir şekilde bir alma işlemi dizisi kullanabilir.

Bir Dizgi kullanılırsa, bir Java sınıfının adı ya da `library(function)` biçiminde bir C programı adı olabilecek, virgülle ayrılmış bir ya da daha çok çıkış tanımlamasından oluşmalıdır.

Daha sonra uygulama, bir MQQueueManager nesnesi yaratmadan önce, Liste ya da Dizgi nesnesini MQEnvironment.channelSendExit alanına atar.

Çıkışlara aktarılan bilgilerin bağlamı yalnızca çıkışların etki alanı içinde yer alıyor. Örneğin, bir Java çıkışı ve bir C çıkışı zincirlenirse, C çıkışı üzerinde Java çıkışı etkisi yoktur.

Çıkış zinciri sınıflarının kullanılması

In versions of WebSphere MQ earlier than Version 7.0, two classes were provided to allow sequences of exits:

- MQSendExit arabirimini gerçekleştirenMQSendExitzinciri
- MQReceiveExit arabirimini gerçekleştirenMQReceiveExitzinciri

Bu sınıfların kullanımı geçerli olmaya devam eder, ancak yeni yöntem tercih edilir. Using the WebSphere MQ Classes for Java interfaces means that your application still has a dependency on `com.ibm.mq.jar` If the new set of interfaces in the `com.ibm.mq.exits` package are used there is no dependency on `com.ibm.mq.jar`.

Bir uygulama çıkış dizisi kullanmak için, bir uygulama, nesnelerin listesini (burada her nesnenin aşağıdakilerden biri olduğu bir liste) oluşturdu:

- MQSendExit arabirimini gerçekleştiren kullanıcı tanımlı bir sınıfın somut örneği (Java 'da yazılan bir gönderme çıkışı için)
- MQExternalSendçıkış sınıfının somut örneği (Java 'da bir gönderme çıkışı için yazılmamış)
- MQSendExitZincir sınıfının somut örneği

Uygulama, bu nesne listesini oluşturucuda değiştirge olarak geçirerek bir MQSendExitChain nesnesi yarattı. Daha sonra, uygulama MQQueueManager nesnesi yaratmadan önce MQSendExitChain nesnesini MQEnvironment.sendExit alanına atayacaktı.

Java için WebSphere MQ sınıflarında kanal sıkıştırması

Bir kanalda akan verilerin sıkıştırılıp açılması, kanalın performansını artırabilir ve ağ trafiğini azaltabilir. IBM WebSphere MQ classes for Java IBM WebSphere MQiçine yerleşik sıkıştırma işlevini kullanın.

IBM WebSphere MQile birlikte verilen işlevi kullanarak, ileti kanallarında ve MQI kanallarında akan verileri sıkıştırabilir ve her iki kanal tipinde de, üstbilgi verilerini ve ileti verilerini birbirinden bağımsız olarak sıkıştırabilirsiniz. Varsayılan olarak, bir kanalda veri sıkıştırılmadı. Kanal sıkıştırması (IBM WebSphere MQiçinde nasıl uygulansa da dahil olmak üzere) tam açıklaması için bkz. Veri sıkıştırması (COMMSG) ve Üstbilgi sıkıştırması (COMMPHDR).

Bir IBM WebSphere MQ classes for Java uygulaması, bir java.util.Collection nesnesi yaratarak bir istemci bağlantısında üstbilgi ya da ileti verileri sıkıştırılması için kullanılabilecek teknikleri belirtir. Her sıkıştırma tekniği, kaynak grubundaki bir Tamsayı nesnesidir ve uygulamanın kaynak grubuna sıkıştırma tekniklerini eklediği sıra, istemci bağlantısı başlatıldığında, sıkıştırma tekniklerinin kuyruk yöneticisiyle kararlaştırıldığı sıradır. Daha sonra uygulama, veri toplama nesnesini hdrCompListe alanına, üstbilgi verileri için ya da msgCompListe alanını, ileti verileri için MQEnvironment sınıfındaki atayabilir. Uygulama hazır olduğunda, bir MQQueueManager nesnesi yaratarak istemci bağlantısını başlatabilir.

Aşağıdaki kod parçaları açıklanan yaklaşımı gösterir. İlk kod parçası, üstbilgi veri sıkıştırmasının nasıl gerçekleştirileceğini gösterir:

```
Collection headerComp = new Vector();
headerComp.add(new Integer(CMQXC.MQCOMPRESS_SYSTEM));
:
MQEnvironment.hdrCompList = headerComp;
:
MQQueueManager qMgr = new MQQueueManager(QM);
```

İkinci kod parçası, ileti veri sıkıştırmasının nasıl gerçekleştirileceğini gösterir:

```
Collection msgComp = new Vector();
msgComp.add(new Integer(CMQXC.MQCOMPRESS_RLE));
msgComp.add(new Integer(CMQXC.MQCOMPRESS_ZLIBHIGH));
:
```

```
MQEnvironment.msgCompList = msgComp;
:
MQQueueManager qMgr = new MQQueueManager(QM);
```

İkinci örnekte, sıkıştırma teknikleri, istemci bağlantısı başlatıldığında ZLIBGHH ' de (SLE) kararlaştırılır. Seçilen sıkıştırma tekniği, MQQueueManager nesnesinin geçerlik süresi boyunca değiştirilemez.

Bir istemci bağlantısında hem istemci, hem de kuyruk yöneticisi tarafından desteklenen üstbilgi ve ileti verilerine ilişkin sıkıştırma teknikleri, bir MQChannelDefinition nesnesinin hdrCompListesi ve msgCompListesi alanlarındaki kaynak grupları olarak bir kanal çıkışa geçirilir. Bir istemci bağlantısında üstbilgi ve ileti verileri sıkıştırılması için kullanılmakta olan gerçek teknikler, MQChannelExit nesnesinin CurHdrSıkıştırması ve CurMsgSıkıştırma alanlarında bir kanal çıkışa geçirilir.

Bir istemci bağlantısında sıkıştırma kullanılıyorsa, veriler, kanal gönderme çıkışları işlendikten sonra, kanal alma çıkışları işlenmeden ve çıkarılmadan önce sıkıştırılır. Gönderme ve alma çıkışlarına aktarılan veriler, bu nedenle sıkıştırılmış durumda.

Sıkıştırma tekniklerini belirtme ve hangi sıkıştırma tekniklerinin kullanılabilir olduğuna ilişkin ek bilgi için Class com.ibm.mq.MQEnvironment ve Interface com.ibm.mq.MQC başlıklı konuya bakın.

IBM WebSphere MQ classes for Javaiçinde bir TCP/IP bağlantısının paylaşılması

Tek bir TCP/IP bağlantısını paylaşmak için bir MQI kanalının birden çok eşgörünümü yapılabilir.

IBM WebSphere MQ classes for Java' ta, tek bir TCP/IP bağlantısını paylaşabilen etkileşimlerin sayısını denetlemek için MQEnvironment.sharingConversations değişkenini kullanıyorsunuz.

SHARECNV özniteliği, bağlantı paylaşımına yönelik en iyi bir çalışma yaklaşımsıdır. Therefore when a SHARECNV value greater than 0 is used with the IBM WebSphere MQ classes for Java it is not guaranteed that a new connection request will always share an already established connection.

Java için WebSphere MQ sınıflarında bağlantı havuzlama

Java içinWebSphere MQ sınıfları, yedek bağlantıların yeniden kullanılmak üzere havuza gönderilmesine olanak tanır.

Java içinWebSphere MQ sınıfları, WebSphere MQ kuyruk yöneticileriyle birden çok bağlantıyla başa geçen uygulamalar için ek destek sağlar. Bir bağlantı artık gerekmediği zaman, onu yok etmek yerine, havuza gönderilebilir ve daha sonra yeniden kullanılabilir. Bu, rasgele kuyruk yöneticilerine dizisel olarak bağlanan uygulamalar ve ara katman yazılımları için önemli bir başarım geliştirmesi sağlayabilir.

WebSphere MQ , varsayılan bir bağlantı havuzu sağlar. Uygulamalar, bu bağlantı havuzunu MQEnvironment sınıfından belirteçler kaydederek ve kayıttan kaldırılarak etkinleştirebilir ya da devre dışı bırakabilir. If the pool is active when WebSphere MQ classes for Java constructs an MQQueueManager object, it searches this default pool and reuses any suitable connection. Bir MQQueueManager.disconnect () çağrısı gerçekleşirse, temeldeki bağlantı havuza geri döndürülür.

Diğer bir seçenek olarak, uygulamalar belirli bir kullanım için bir MQSimpleConnectionManager bağlantı havuzu oluşturabilirler. Daha sonra, uygulama bir MQQueueManager nesnesinin yapımı sırasında havuzu belirtebilir ya da varsayılan bağlantı havuzu olarak kullanılmak üzere bu havuzu MQEnvironment 'a geçirebilir.

Bağlantıların çok fazla kaynak kullanmasını önlemek için, bir MQSimpleConnectionYöneticisi nesnesinin işleyebileceği toplam bağlantı sayısını sınırlayabilir ve bağlantı havuzunun büyüklüğünü sınırlayabilirsiniz. Bir JVM içindeki bağlantılar için çakışan talepler varsa, bu sınırların ayarlanması yararlı olur.

Varsayılan olarak, getMaxConnections () yöntemi sıfır değerini döndürür; bu da, MQSimpleConnectionManager nesnesinin işleyebileceği bağlantı sayısı için bir sınır olmadığı anlamına gelir. setMaxConnections () yöntemini kullanarak bir sınır ayarlayabilirsiniz. Bir sınır ayarlayıp sınıra ulaşılırsa, daha fazla bağlantı için bir istek, MQRC_MAX_CONNS_IMlim_REACHED neden koduyla bir MQException yayınlanmasına neden olabilir.

Java için WebSphere MQ sınıflarında varsayılan bağlantı havuzunu denetleme

Bu örnek, varsayılan bağlantı havuzunun nasıl kullanılacağını gösterir.

Aşağıdaki örnek uygulamayı göz önünde bulundurun: MQApp1:

```
import com.ibm.mq.*;
public class MQApp1
{
      public static void main(String[] args) throws MQException
      {
         for (int i=0; i<args.length; i++) {
            MQQueueManager qmgr=new MQQueueManager(args[i]);
            :
            : (do something with qmgr)
            :
            qmgr.disconnect();
         }
      }
}
```

MQApp1 , komut satırından yerel kuyruk yöneticilerinin listesini alır, her bir sırayla bağlanır ve bazı işlemler gerçekleştirir. Ancak, komut satırı aynı kuyruk yöneticisini birçok kez listelediğinde, bu bağlantıyı yalnızca bir kez bağlamak ve bu bağlantıyı birçok kez yeniden kullanmak daha verimli olur.

Java için WebSphere MQ class for Java, bunu yapmak için kullanabileceğiniz varsayılan bir bağlantı havuzu sağlar. Havuzu etkinleştirmek için, MQEnvironment.addConnectionPoolToken() yöntemlerinden birini kullanın. Havuzu geçersiz kılmak için MQEnvironment.removeConnectionPoolToken() öğesini kullanın.

Aşağıdaki örnek uygulama (MQApp2), işlevsel olarak MQApp1ile aynı, ancak her kuyruk yöneticisine yalnızca bir kez bağlanır.

```
import com.ibm.mq.*;
public class MQApp2
{
      public static void main(String[] args) throws MQException
      {
         MQPoolToken token=MQEnvironment.addConnectionPoolToken();

         for (int i=0; i<args.length; i++) {
            MQQueueManager qmgr=new MQQueueManager(args[i]);
            :
            : (do something with qmgr)
            :
            qmgr.disconnect();
         }

         MQEnvironment.removeConnectionPoolToken(token);

      }
}
```

İlk kalın çizgi, MQEnvironment içeren bir MQPoolToken nesnesini kaydettirerek varsayılan bağlantı havuzunu etkinleştirir.

MQQueueManager oluşturucusu, bu havuzu uygun bir bağlantı için arar ve var olan bir bağlantı bulamazsa kuyruk yöneticiyle bağlantı yaratır. qmgr.disconnect() çağrısı, daha sonra yeniden kullanılmak üzere havuzla bağlantıyı döndürür. Bu API çağrıları örnek uygulamayla (MQApp1) aynıdır.

Vurgulu görüntülenen ikinci satır, havuzda saklanan kuyruk yöneticisi bağlantılarını yok eden varsayılan bağlantı havuzunu devre dışı bırakır. Bunun nedeni, uygulamanın havuzdaki canlı kuyruk yöneticisi bağlantılarıyla sona erdirileceği için önemlidir. Bu durum, kuyruk yöneticisi günlüklerinde görüntülenecek hatalara neden olabilir.

Bir uygulama bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, MQQueueManager oluşturucusu, önce uygun bir istemci bağlantı kanalı tanımlaması için çizelgeyi arar. Bir bağlantı bulunursa, oluşturucu, kanal için kullanılabilecek bir bağlantı için varsayılan bağlantı havuzunu arar. Oluşturucu havuzda uygun bir bağlantı bulamazsa, sonraki uygun istemci bağlantı kanalı tanımlaması için istemci kanal tanımlama çizelgesinde arama yapar ve daha önce açıklandığı gibi

devam eder. Oluşturucu, istemci kanal tanımlama çizelgesini aramasını tamamlarsa ve havuzda uygun bir bağlantı bulamazsa, oluşturucu, çizelgenin ikinci bir aramasını başlatır. Bu arama sırasında oluşturucu, her bir uygun istemci bağlantı kanalı tanımlaması için yeni bir bağlantı yaratmayı dener ve bu bağlantı, yaratmayı başardığı ilk bağlantıyı kullanır.

Varsayılan bağlantı havuzu, en çok on kullanılmamış bağlantıyı saklar ve kullanılmayan bağlantıları en fazla beş dakika etkin tutar. Uygulama bunu değiştirebilir (ayrıntılar için bkz. “Java için WebSphere MQ sınıflarında farklı bir bağlantı havuzu sağlama” sayfa 668).

Bir MQPoolTokensağlamak için MQEnvironment 'ı kullanmak yerine, uygulama kendi yapısını oluşturabilir:

```
MQPoolToken token=new MQPoolToken();
MQEnvironment.addConnectionPoolToken(token);
```

Some applications or middleware vendors provide subclasses of MQPoolToken in order to pass information to a custom connection pool. Bağlantı havuzuna ek bilgi aktarılabilmesi için, bunlar yaratılabilir ve bu şekilde addConnectionPoolToken() yoluna geçebilirler.

Java için WebSphere MQ sınıflarında varsayılan bağlantı havuzu ve birden çok bileşen

This example shows how to add or remove MQPoolTokens from a static set of registered MQPoolToken objects.

MQEnvironment, kayıtlı bir MQPoolToken nesnesi kümesini içerir. Bu kümeden MQPoolTokens eklemek ya da kaldırmak için aşağıdaki yöntemleri kullanın:

- MQEnvironment.addConnectionPoolToken()
- MQEnvironment.removeConnectionPoolToken()

Bir uygulama, bağımsız olarak var olan birçok bileşenden oluşabilir ve kuyruk yöneticisi kullanarak iş gerçekleştirir. Böyle bir uygulamada, her bileşen, yaşam süresi için MQEnvironment kümesine bir MQPoolToken eklemelidir.

Örneğin, örnek uygulama MQApp3 on iş parçacığı yaratır ve her birini başlatır. Her iş parçacığı kendi MQPoolToken' ı kaydeder, süre uzunluğunu bekler ve kuyruk yöneticisine bağlanır. İş parçacığı bağlantısını kestikten sonra, kendi MQPoolToken' ı kaldırır.

Varsayılan bağlantı havuzu, MQPoolTokenskümesinde en az bir simge olduğunda etkin kalır; bu nedenle, bu uygulama bu uygulamanın süresi boyunca etkin kalır. Uygulamanın, iş parçacıklarının genel denetiminde ana nesne tutması gerekmez.

```
import com.ibm.mq.*;
public class MQApp3
{
      public static void main(String[] args)
      {
         for (int i=0; i<10; i++) {
            MQApp3_Thread thread=new MQApp3_Thread(i*60000);
            thread.start();
         }
      }
}

class MQApp3_Thread extends Thread
{
      long time;

      public MQApp3_Thread(long time)
      {
         this.time=time;
      }

      public synchronized void run()
      {
         MQPoolToken token=MQEnvironment.addConnectionPoolToken();
         try {
            wait(time);
            MQQueueManager qmgr=new MQQueueManager("my.qmgr.1");
            :
```

```
            : (do something with qmgr)
            :
            qmgr.disconnect();
         }
         catch (MQException mqe) {System.err.println("Error occurred!");}
         catch (InterruptedException ie) {}

         MQEnvironment.removeConnectionPoolToken(token);
      }
}
```

Java için WebSphere MQ sınıflarında farklı bir bağlantı havuzu sağlama

Bu örnekte, farklı bir bağlantı havuzu sağlamak için **com.ibm.mq.MQSimpleConnectionManager** sınıfının nasıl kullanılacağı gösterilmektedir.

Bu sınıf, bağlantı havuzlama için temel olanaklar sağlar ve uygulamalar, havuzun davranışını uyarlamak için bu sınıfı kullanabilir.

Bir örnek oluşturulduktan sonra, MQQueueManager oluşturucuda bir MQSimpleConnectionManager belirtilebilir. Daha sonra, oluşturulan MQQueueManager' in temelini oluşturan bağlantıyı yöneten MQSimpleConnectionManager. MQSimpleConnectionManager uygun bir havuza yollanmış bağlantı içeriyorsa, o bağlantı yeniden kullanılır ve MQQueueManager.disconnect () çağrısından sonra MQSimpleConnectionManager 'a döndürülür.

Aşağıdaki kod parçası bu davranışı gösterir:

```
MQSimpleConnectionManager myConnMan=new MQSimpleConnectionManager();
myConnMan.setActive(MQSimpleConnectionManager.MODE_ACTIVE);
MQQueueManager qmgr=new MQQueueManager("my.qmgr.1", myConnMan);
:
: (do something with qmgr)
:
qmgr.disconnect();

MQQueueManager qmgr2=new MQQueueManager("my.qmgr.1", myConnMan);
:
: (do something with qmgr2)
:
qmgr2.disconnect();
myConnMan.setActive(MQSimpleConnectionManager.MODE_INACTIVE);
```

İlk MQQueueManager oluşturucusu sırasında oluşan bağlantı, qmgr.disconnect() çağrısından sonra myConnMan içinde saklanır. The connection is then reused during the second call to the MQQueueManager constructor.

İkinci satır MQSimpleConnectionManager 'ı etkinleştirir. The last line disables MQSimpleConnectionManager, destroying any connections held in the pool. Bir MQSimpleConnectionManager, bu bölümde daha sonra açıklanan MODE_AUTO ' da varsayılan değer olarak.

MQSimpleConnectionYöneticisi, en son kullanılan bir temelde bağlantı ayırır ve bağlantıları en az kullanılan en az kullanılan bir temelde yok eder. Varsayılan değer olarak, beş dakika kullanılmadıysa ya da havuzda kullanılmayan on 'dan fazla bağlantı varsa, bağlantı yok edilir. Bu değerleri değiştirmek için MQSimpleConnectionManager.setTimeout() öğesini çağırarak değiştirebilirsiniz.

MQQueueManager oluşturucuda herhangi bir Bağlantı Yöneticisi sağlanmadığında kullanılacak varsayılan bağlantı havuzu olarak kullanılacak bir MQSimpleConnectionManager 'ı da ayarlayabilirsiniz.

Aşağıdaki uygulama bunu göstermektedir:

```
import com.ibm.mq.*;
public class MQApp4
{
      public static void main(String []args)
      {
         MQSimpleConnectionManager myConnMan=new MQSimpleConnectionManager();
         myConnMan.setActive(MQSimpleConnectionManager.MODE_AUTO);
         myConnMan.setTimeout(3600000);
         myConnMan.setMaxConnections(75);
```

```
            myConnMan.setMaxUnusedConnections(50);
            MQEnvironment.setDefaultConnectionManager(myConnMan);
            MQApp3.main(args);
        }
}
```

Kalın çizgiler bir MQSimpleConnectionManager nesnesi yaratabilir ve yapılandırabilr. Yapılandırma şunları yapar:

- Bir saat kullanılmayan bağlantıları sona erdirir
- myConnMan tarafından yönetilen bağlantı sayısını 75 olarak sınırlar.
- Havuzdaki kullanılmayan bağlantı sayısını 50 ile sınırlar
- MODE_AUTO ' yı belirler; varsayılan değer bu. Bu, havuzun yalnızca varsayılan bağlantı yöneticisi olması durumunda etkin olduğu ve MQEnvironment tarafından tutulan MQPoolTokens kümesinde en az bir simge olması anlamına gelir.

Yeni MQSimpleConnectionManager, varsayılan bağlantı yöneticisi olarak ayarlanır.

In the last line, the application calls MQApp3.main(). This runs a number of threads, where each thread uses WebSphere MQ independently. Bu iş parçacıkları, bağlantı kurdukları sırada myConn(MyConn) olanağını kullanır.

Supplying your own ConnectionManager for WebSphere MQ classes for Java

Java için WebSphere MQ sınıfları, javax.resource.spi.ConnectionManager ' un somutlamalarına izin veren Java EE Connector Architecture mimarisinin bir kısmi somutlamasını sağlar.

Uygulamalar ve ara katman yazılımı sağlayıcıları, bağlantı havuzlarına alternatif uygulamalar sağlayabilir. Java için WebSphere MQ sınıfları, Java EE bağlayıcı mimarisinin kısmi bir somutlamasını sağlar. **javax.resource.spi.ConnectionManager** ' ın somutlamaları varsayılan Bağlantı Yöneticisi olarak kullanılabilir ya da MQQueueManager oluşturucuda belirtilebilir.

Java için WebSphere MQ sınıfları, Java EE Bağlayıcı Mimarisinin Bağlantı Yönetimi sözleşmesiyle uyumludur. Bu bölümü, Java EE Bağlayıcı Mimarisinin Bağlantı Yönetimi (Connection Management) sözleşmesiyle birlikte okuyun (https://java.sun.comadresindeki Sun 'ın Java Web sitesine bakın).

ConnectionManager arabirimi yalnızca bir yöntem tanımlar:

```
package javax.resource.spi;
public interface ConnectionManager {
      Object allocateConnection(ManagedConnectionFactory mcf,
                                ConnectionRequestInfo cxRequestInfo);
}
```

MQQueueManager oluşturucusu, ilgili ConnectionManager'deki allocateConnection ' u çağırır. Gereken bağlantıyı tanımlamak için, uygun ManagedConnectionFactory ve ConnectionRequestInfo somutlamalarını parametre olarak geçirir.

ConnectionManager , özdeş ManagedConnectionFactory ve ConnectionRequestInfo nesneleriyle oluşturulmuş bir javax.resource.spi.ManagedConnection nesnesi için havuzu arar. If the ConnectionManager finds any suitable ManagedConnection objects, it creates a java.util.Set that contains the candidate ManagedConnections. Daha sonra, ConnectionManager aşağıdakileri çağırır:

```
ManagedConnection mc=mcf.matchManagedConnections(connectionSet, subject,
cxRequestInfo);
```

ManagedConnectionFactory 'nin WebSphere MQ uygulaması, konu parametresini yoksayar. Bu yöntem, setten uygun bir ManagedConnection seçer ve döndürür ya da uygun bir ManagedConnectionbulamazsa boş değer döndürür. Havuzda uygun bir ManagedConnection yoksa, ConnectionManager aşağıdaki özellikleri kullanarak bir tane yaratabilir:

```
ManagedConnection mc=mcf.createManagedConnection(subject, cxRequestInfo);
```

Yine, konu değiştirgesi yoksayılır. Bu yöntem, bir WebSphere MQ kuyruk yöneticisine bağlanır ve yeni kurulan bağlantıyı gösteren javax.resource.spi.ManagedConnection somutlamasını döndürür. ConnectionManager bir ManagedConnection (havuzdan ya da yeni oluşturulduktan sonra) edindikten sonra, aşağıdaki özellikleri kullanarak bir bağlantı tanıtıcısı yaratır:

```
Object handle=mc.getConnection(subject, cxRequestInfo);
```

Bu bağlantı tanıtıcısı allocateConnection() içinden döndürülebilmektedir.

A ConnectionManager must register an interest in the ManagedConnection through:

```
mc.addConnectionEventListener()
```

Bağlantıda önemli bir hata oluşursa ya da MQQueueManager.disconnect () çağrıldığında ConnectionEventListener uyarılır. MQQueueManager.disconnect () çağrıldığında, ConnectionEventDinleyicisi aşağıdakilerden birini yapabilir:

- mc.cleanup() çağrısını kullanarak ManagedConnection 'ı sıfırlayın ve daha sonra, ManagedConnection ' ı havuza geri gönderin
- mc.destroy() çağrısını kullanarak ManagedConnection ' ı yok edin

ConnectionManager varsayılan ConnectionManagerise, MQEnvironment yönetimli MQPoolTokenskümesinin durumuna da ilgi kaydedebilir. Bunu yapmak için önce bir MQPoolServices nesnesi oluşturun ve ardından MQPoolServices nesnesiyle bir MQPoolServicesEventListener nesnesini kaydettirin:

```
MQPoolServices mqps=new MQPoolServices();
mqps.addMQPoolServicesEventListener(listener);
```

Bir MQPoolToken eklendiğinde ya da kümeden kaldırıldığında ya da varsayılan ConnectionManager değişiklikleri olduğunda dinleyici bilgilendirilir. MQPoolServices nesnesi ayrıca, MQPoolTokenskümesinin geçerli boyutunu sorgulamak için bir yol da sağlar.

Java için WebSphere MQ sınıfları kullanılarak JTA/JDBC eşgüdümü

WebSphere MQ classes for Java supports the MQQueueManager.begin() method, which allows WebSphere MQ to act as a coordinator for a database which provides a JDBC type 2 or JDBC type 4 compliant driver.

Bu destek tüm altyapılarda kullanılamaz. Hangi platformların JDBC koordinasyonunu desteklediğini denetlemek için, bkz. https://www.ibm.com/software/integration/wmq/requirements/.

XA-JTA desteğini kullanmak için özel JTA anahtar kitaplığını kullanmanız gerekir. Bu kitaplığın kullanılmasına ilişkin yöntem, Windows ya da diğer altyapılardan birini kullanıp kullanmamanıza bağlı olarak değişir.

Windowsüzerinde JTA/JDBC koordinasyonunun yapılandırılması

XA kitaplığı, `jdbcxxx.dll`biçiminde bir DLL adı olarak sağlanır.

V 7.5.0.7 Sağlanan `jdbcora12.dll` , bir IBM WebSphere MQ Windows sunucusu kuruluşu için Oracle 12Cile uyumluluk sağlar.

Windows sistemlerinde XA kitaplığı tam bir DLL olarak sağlanır. Bu DLL ' in adı `jdbcxxx.dll` olur; burada `xxx` , anahtar kitaplığının derlendiği veritabanını gösterir. Bu kitaplık, Java kuruluşuna ilişkin IBM WebSphere MQ sınıflarınızın `java\lib\jdbc` ya da `java\lib64\jdbc` dizininde yer almaktadır. Anahtar yükleme dosyası olarak da tanımlanan XA kitaplığını kuyruk yöneticisine bildirmeniz gerekir. IBM WebSphere MQ Explorer' yi kullanın. XA kaynak yöneticisi altında, kuyruk yöneticisi özellikleri panosunda anahtar yükleme dosyasının ayrıntılarını belirtin. Kitaplığın adını yalnızca siz vermelisiniz. Örneğin:

Bir Db2 veritabanı için, SwitchFile alanını şu şekilde ayarlayın: dbcdb2

Bir Oracle veritabanı için, SwitchFile alanını şu şekilde ayarlayın: `jdbcora`

Pencerelerdışındaki platformlarda JTA/JDBC eşgüdümü yapılandırılması

Nesne dosyaları sağlanır. Sağlanan makefile 'ı kullanarak uygun olanını bağlayın ve yapılandırma dosyasını kullanarak kuyruk yöneticisine bildirin.

Her veritabanı yönetim sistemi için, WebSphere MQ iki nesne dosyası sağlar. 32 bit anahtar kitaplığı yaratmak için bir nesne dosyasını bağlamanız ve 64 bit anahtar kitaplığı yaratmak üzere diğer nesne dosyasını bağlamanız gerekir. DB2için, her nesne dosyasının adı jdbcdb2.o 'dır ve Oracleiçin her bir nesne dosyasının adı jdbcora.o' dur.

Her nesne dosyasını, WebSphere MQile sağlanan uygun makefile kullanarak bağlamanız gerekir. Anahtar kitaplığı, farklı sistemlerde farklı konumlarda saklanabilen diğer kitaplıkları gerektirir. Ancak, anahtar kitaplığı bir setuid ortamında çalışan kuyruk yöneticisi tarafından yüklendiğinden, bir anahtar kitaplığı bu kitaplıkları bulmak için kitaplık yolu ortam değişkenini kullanamaz. Bu nedenle, sağlanan makefile, bir anahtar kitaplığının bu kitaplıkların tam olarak nitelenmiş yol adlarını içermesini sağlar.

Bir anahtar kitaplığı oluşturmak için, aşağıdaki biçimi kullanarak bir **make** komutu girin. 32 bit anahtar kitaplığı yaratmak için, komutu WebSphere MQ kurulumunuzun /java/lib/jdbc dizinine girin. 64 bit anahtar kitaplığı oluşturmak için, komutu /java/lib64/jdbc dizinine girin.

```
make DBMS
```

Burada *DBMS* , anahtar kitaplığını oluşturduğunuz veritabanı yönetim sistemidir. Geçerli değerler, DB2 için db2 ve Oracleiçin `oracle` değerleridir.

Aşağıda bir **make** komutu örneği yer almaktadır:

```
make db2
```

Aşağıdaki noktalara dikkat edin:

- 32 bit uygulamaları çalıştırmak için, kullanmakta olduğunuz her bir veritabanı yönetim sistemi için hem 32 bit hem de 64 bit anahtar kitaplığı yaratmalısınız. 64 bit uygulamaları çalıştırmak için, yalnızca 64 bit anahtar kitaplığı yaratmanız gerekir. DB2için, her anahtar kitaplığının adı jdbcdb2 'dir ve Oracleiçin her anahtar kitaplığının adı jdbcora' dır. Makefiles, 32 bit ve 64 bit anahtar kitaplıklarının farklı WebSphere MQ dizinlerinde depolandığından emin olur. 32 bit anahtar kitaplığı /java/lib/jdbc dizininde saklanır ve bir 64 bit anahtar kitaplığı /java/lib64/jdbc dizininde saklanır.
- Oracle 'ı bir sistemin herhangi bir yerinde kurabileceğiniz için, makefiles, Oracle ' ın kurulu olduğu yeri bulmak için ORACLE_HOME ortam değişkenini kullanır.

After you have created the switch libraries for DB2, Oracle, or both, you must declare them to your queue manager. Kuyruk yöneticisi yapılandırma dosyası (qm.ini), DB2 ya da Oracle veritabanlarına ilişkin XAResourceManager stanzaları içeriyorsa, her bir Stanza içindeki SwitchFile girdisini aşağıdakilerden biri ile değiştirmeniz gerekir:

DB2 veritabanı için

```
SwitchFile=jdbcdb2
```

Oracle veritabanı için

```
SwitchFile=jdbcora
```

32 bit ya da 64 bit anahtar kitaplığının tam olarak nitelenmiş yol adını belirtmeyin. Yalnızca kitaplığın adını belirtin.

If the queue manager configuration file does not already contain XAResourceManager stanzas for DB2 or Oracle databases, or if you want to add additional XAResourceManager stanzas, see Yönetme for information about how to construct an XAResourceManager stanza. Ancak, yeni bir XAResourceManager stanza içindeki her SwitchFile girişi, tam olarak DB2 ya da Oracle veritabanı için anlatıldığı gibi olmalıdır. You must also include the entry `ThreadOfControl=PROCESS`.

Kuyruk yöneticisi yapılanış kütüğünü güncelledikten ve tüm uygun veritabanı ortam değişkenlerinin ayarlandığından emin olduktan sonra, kuyruk yöneticisini yeniden başlatabilirsiniz.

JTA/JDBC koordinasyonunun kullanılması

API çağrılarınızı sağlanan örnekte olduğu gibi kodlayın.

Bir kullanıcı uygulaması için API çağrılarının temel sırası şöyledir:

```
  qMgr = new MQQueueManager("QM1")
  Connection con = qMgr.getJDBCConnection( xads );
  qMgr.begin()

 < Perform MQ and DB operations to be grouped in a unit of work >

  qMgr.commit() or qMgr.backout();
  con.close()
  qMgr.disconnect()
```

getJDBCConnection çağrısındakixads , bağlanmak için veritabanının ayrıntılarını tanımlayan XADataSource arabiriminin veritabanına özgü bir somutlamasını içerir. getJDBCConnection' a geçmek üzere uygun bir XADataSource nesnesinin nasıl yaratılacağı hakkında bilgi almak için veritabanınıza ilişkin belgelere bakın.

You must also update your class path with the appropriate database-specific jar files for performing JDBC work.

Birden çok veritabanına bağlanmanız gerekiyorsa, birçok farklı bağlantıda işlemi gerçekleştirmek için getJDBCConnection çağrısını birkaç kez aramalısınız.

There are two forms of the getJDBCConnection, reflecting the two forms of XADataSource.getXAConnection:

```
  public java.sql.Connection getJDBCConnection(javax.sql.XADataSource xads)
    throws MQException, SQLException, Exception

  public java.sql.Connection getJDBCConnection(XADataSource dataSource,
                                               String userid, String password)
    throws MQException, SQLException, Exception
```

Bu yöntemler, JTA işlevlerini kullanmayan müşteriler için JVM doğrulayıcındaki sorunları önlemek için throws yantümcelerinde kural dışı durum bildirmektedir. Atılan gerçek kural dışı durum javax.transaction.xa.XAException , daha önce gerektirmeyen programlar için sınıf yoluna jta.jar dosyasının eklenmesini gerektirir.

JTA/JDBC desteğini kullanmak için, uygulamanıza aşağıdaki deyimi eklemelisiniz:

```
MQEnvironment.properties.put(CMQC.THREAD_AFFINITY_PROPERTY, new Boolean(true));
```

JTA/JDBC eşgüdümü ile ilgili bilinen sorunlar ve sınırlamalar

Bazı sorunlar ve JTA/JDBC desteği sınırlamaları vardır. Bazıları, kullanılan veritabanı yönetim sistemine bağlıdır.

Bu destek, JDBC sürücülerini çağırdığı için, bu JDBC sürücülerinin somutlaması, sistem işleyişi üzerinde önemli bir etkiye sahip olabilir. Özellikle, test edilen JDBC sürücüleri, bir uygulama çalışırken veritabanı kapatıldığında farklı davranır. **Her zaman** , bir veritabanını açık bağlantıları tutan uygulamalar varken, bir veritabanını aniden kapatmaktan kaçının.

Birden çok XAResourceManager stanzası

Bir kuyruk yöneticisi yapılanış kütüğünde qm.iniolan birden çok XAResourceManager stanza kullanımı desteklenmez. Birinciden başka herhangi bir XAResourceManager kısmı yok sayılır.

DB2

Bazen DB2 , bir SQL0805N hatası döndürür. Bu sorun şu CLP komutlarıyla çözülebilir:

```
DB2 bind @db2cli.lst blocking all grant public
```

Ek bilgi için DB2 belgelerine bakın.

XAResourceManager stanza, ThreadOfControl=PROCESS kullanacak şekilde yapılandırılmalıdır. For DB2 version 8.1 and higher this does not match the default thread of control setting for DB2, so toc=p must be specified in the XA Open String. JTA/JDBC eşgüdümü ile DB2 için örnek bir XAResourceManager stanza şu şekildir:

```
XAResourceManager:
     Name=jdbcdb2
     SwitchFile=jdbcdb2
     XAOpenString=uid=userid,db=dbalias,pwd=password,toc=p
     ThreadOfControl=PROCESS
```

Bu, JTA/JDBC eşgüdümü kullanan Java uygulamalarının çok iş parçacıklı kendilerinden oluşmasını önlemez.

Oracle

MQQueueManager.disconnect () işleminden sonra JDBC Connection.close() yöntemi bir SQL kural dışı durumu (SQL kural dışı durumu) yaratır. Either call Connection.close() before MQQueueManager.disconnect(), or omit the call to Connection.close().

Java için WebSphere MQ sınıflarında SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) desteği

Java istemcisi uygulamaları için WebSphere MQ sınıfları SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) şifrelemesini destekler. SSL şifrelemesini kullanmak için bir JSSE sağlayıcısına gereksinim duyarsınız.

WebSphere MQ classes for Java client applications using TRANSPORT(CLIENT) support Secure Sockets Layer (SSL) encryption. SSL, iletişim şifrelemesi, kimlik doğrulaması ve ileti bütünlüğü sağlar. Bu, genellikle İnternet üzerinde ya da bir intranet içinde herhangi iki eş arasında iletişim sağlamak için kullanılır.

Java için WebSphere MQ sınıfları, SSL şifrelemesini işlemek için Java Secure Socket Extension (JSSE) olanağını kullanır ve JSSE sağlayıcısını gerektirir. JSE v1.4 JVM ' ler yerleşik bir JSSE sağlayıcısına sahip. Sertifikaların nasıl yönetileceği ve saklanabileceği ile ilgili ayrıntılar sağlayıcıdan sağlayıcıya göre değişebilir. Bu konuda bilgi almak için, JSSE sağlayıcısının belgelerine bakın.

Bu bölümde, JSSE sağlayıcınızın doğru bir şekilde kurulmuş ve yapılandırılmış olduğu ve JSSE sağlayıcınız için uygun sertifikaların kurulmuş ve kullanılabilir durumda olduğu varsayılır.

Java istemcisi uygulaması için WebSphere MQ sınıflarınız bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, “Using a client channel definition table with IBM WebSphere MQ classes for Java” sayfa 645başlıklı konuya bakın.

Enabling SSL in IBM WebSphere MQ classes for Java

SSL ' i etkinleştirmek için bir CipherSuitebelirtmenizi sağlar. Bir CipherSuitebelirtmenin iki yolu vardır.

SSL yalnızca istemci bağlantıları için desteklenir. SSL ' yi etkinleştirmek için, kuyruk yöneticisiyle iletişim kurarken kullanılacak CipherSuite değerini belirlemeniz gerekir; bu CipherSuite , hedef kanaldaki CipherSpec ayarına uygun olmalıdır. Buna ek olarak, JSSE sağlayıcınız tarafından desteklenen CipherSuite ' in de desteklenmesi gerekir. Ancak, CipherSuites , CipherSpecs ' dan farklıdır ve farklı adlara sahiptir. “Java için WebSphere MQ sınıflarında SSL CipherSpecs ve CipherSuites” sayfa 678 contains a table mapping the CipherSpecs supported by IBM WebSphere MQ to their equivalent CipherSuites as known to JSSE.

SSL 'yi etkinleştirmek için, MQEnvironment 'ın sslCipherSuite statik üye değişkenini kullanarak CipherSuite ' i belirtin. Aşağıdaki örnek, SECURE.SVRCONN.CHANNEL ile RC4_MD5_EXPORT:ile SSL gerektirecek şekilde ayarlanmış olan KANAL:

```
MQEnvironment.hostname       = "your_hostname";
MQEnvironment.channel        = "SECURE.SVRCONN.CHANNEL";
MQEnvironment.sslCipherSuite = "SSL_RSA_EXPORT_WITH_RC4_40_MD5";
MQQueueManager qmgr = new MQQueueManager("your_Q_manager");
```

Kanalda RC4_MD5_EXPORTiçin bir CipherSpec değeri olmasına rağmen, Java uygulaması SSL_RSA_EXPORT_WITH_RC4_40_MD5için bir CipherSuite belirtmelidir. See “Java için WebSphere MQ sınıflarında SSL CipherSpecs ve CipherSuites” sayfa 678 for a list of mappings between CipherSpecs and CipherSuites.

Bir uygulama ayrıca CMQC.SSL_CIPHER_SUITE_PROPERTYortam özelliğini ayarlayarak bir CipherSuite de belirtebilir.

Diğer bir seçenek olarak, İstemci Kanal Tanımlama Çizelgesi 'ni (CCDT) kullanın. Daha fazla bilgi için bkz. “Using a client channel definition table with IBM WebSphere MQ classes for Java” sayfa 645

If you require a client connection to use a CipherSuite that is supported by the IBM Java JSSE FIPS provider (IBMJSSEFIPS), an application can set the sslFipsRequired field in the MQEnvironment class to `true`. Diğer bir seçenek olarak, uygulama CMQC.SSL_FIPS_REQUIRED_PROPERTYortam özelliğini ayarlayabilir. Varsayılan değer `false`değeridir. Bu, bir istemci bağlantısının IBM WebSphere MQtarafından desteklenen herhangi bir CipherSuite ' i kullanabileceğini belirtir.

Bir uygulama birden çok istemci bağlantısı kullanıyorsa, uygulama ilk istemci bağlantısını yarattığında kullanılan sslFipsGerekli alanının değeri, uygulama sonraki istemci bağlantısı yarattığında kullanılan değeri belirler. Bu nedenle, uygulama sonraki bir istemci bağlantısı yarattığında, sslFipsRequired (Gerekli) alanının değeri yoksayılır. sslFipsZorunlu alanı için farklı bir değer kullanmak istiyorsanız uygulamayı yeniden başlatmalısınız.

SSL ' yi başarıyla kullanarak bağlanmak için, JSSE güvenli deposunun, kuyruk yöneticisi tarafından sunulan sertifikana doğrulanabilmesi için sertifika yetkilisi kök sertifikalarıyla ayarlanması gerekir. Benzer şekilde, SVRCONN kanalında SSLClientAuth MQSSL_CLIENT_AUTH_REQUIRED olarak ayarlandıysa, JSSE anahtar deposu, kuyruk yöneticisi tarafından güvenilen bir tanıtıcı sertifika içermelidir.

İlgili başvurular

UNIX, Linux ve Windows için Federal Bilgi İşleme Standartları (FIPS)

IBM WebSphere MQ classes for Java' da kuyruk yöneticisinin ayırt edici adını kullanma

Kuyruk yöneticisi kendisini ayırt edici bir ad (DN) içeren bir SSL sertifikası kullanarak tanımlar. Bir IBM WebSphere MQ classes for Java istemcisi uygulaması, doğru kuyruk yöneticisiyle iletişim kurduğundan emin olmak için bu DN ' yi kullanabilir.

MQEnvironment 'ın sslPeerAd değişkeni kullanılarak bir DN örünü belirtildi. Örneğin:

```
MQEnvironment.sslPeerName = "CN=QMGR.*, OU=IBM, OU=WEBSPHERE";
```

Ancak kuyruk yöneticisi, QMGR. ' un başında bir Common Name değeri olan bir sertifika sunarsa, bağlantının başarılı olmasına izin verir. İlk olarak IBM ve ikinci WebSphereolmak üzere en az iki Organizatif Birim adı olmalıdır.

sslPeerAdı ayarlandıysa, bağlantılar yalnızca geçerli bir örüntü olarak ayarlandıysa ve kuyruk yöneticisi eşleşen bir sertifika sunarsa başarılı olur.

Bir uygulama ayrıca CMQC.SSL_PEER_NAME_PROPERTYortam özelliğini ayarlayarak, kuyruk yöneticisinin ayırt edici adını da belirtebilir. Ayırt edici adlara ilişkin ek bilgi için Ayırt edici adlarbaşlıklı konuya bakın.

Using certificate revocation lists in IBM WebSphere MQ classes for Java

java.security.cert.CertStore sınıfı aracılığıyla kullanılacak sertifika iptal listelerini belirtin. IBM WebSphere MQ classes for Java Bu durumda, sertifikaları belirlenen CRL ' ye göre denetler.

Sertifika iptal listesi (CRL), sertifika yetkilisi ya da yerel kuruluş tarafından iptal edilen bir sertifikalar kümesidir. CRL ' ler genellikle LDAP sunucularında barındırılır. Java 2 v1.4ile, bağlantı sırasında bir CRL sunucusu belirtilebilir ve kuyruk yöneticisi tarafından sunulan sertifika, bağlantıya izin verilmeden önce CRL ' ye yönelik olarak denetlenir. Sertifika iptal listelerine ve IBM WebSphere MQ' a ilişkin daha fazla bilgi için bkz. Sertifika İptal Listeleri ve Yetki İptal Listeleriyle Çalışma ve Java için WebSphere MQ sınıflarına ve JMS için WebSphere MQ sınıflarına CRL ve ARL ' lere erişme.

Not: Bir LDAP sunucusunda barındırılan bir CRL ile başarılı bir şekilde CertStore kullanmak için, Java Software Development Kit (SDK) ürününüzün CRL ile uyumlu olduğundan emin olun. Bazı SDK 'lar, CRL' nin, LDAP v2için bir şema tanımlayan RFC 2587 'ye uymasını gerektirir. Çoğu LDAP v3 sunucusu, bunun yerine RFC 2256 'yı kullanır.

Kullanılacak CRL ' ler java.security.cert.CertStore sınıfı aracılığıyla belirtilir. CertStoreyönetim ortamlarının nasıl edinileceği ile ilgili ayrıntılı bilgi için bu sınıftaki belgelere bakın. LDAP sunucusuna dayalı olarak bir CertStore yaratmak için, önce bir LDAPCertStoreDeğiştirgeleri eşgörünümü yaratın; kullanılacak sunucu ve kapı ayarlarıyla kullanıma hazırlandı. Örneğin:

```
import java.security.cert.*;
CertStoreParameters csp = new LDAPCertStoreParameters("crl_server", 389);
```

Bir CertStoreDeğiştirgeleri eşgörünümü yarattıktan sonra, LDAP tipinde bir CertStore yaratmak için CertStore ' da durağan oluşturucuyu kullanın:

```
CertStore cs = CertStore.getInstance("LDAP", csp);
```

Diğer CertStore tipleri (örneğin, Derlem) de desteklenir. Genellikle, yedeklilik sağlamak için aynı CRL bilgileriyle birlikte ayarlanan birden çok CRL sunucusu vardır. Bu CRL sunucularının her biri için bir CertStore nesnesi bulunduğunda, bunları uygun bir Toplamaya yerleştirin. Aşağıdaki örnekte, ArrayList(ArrayList) içine yerleştirilen CertStore nesneleri gösterilmektedir:

```
import java.util.ArrayList;
Collection crls = new ArrayList();
crls.add(cs);
```

CRL denetimini etkinleştirmek için bağlanmadan önce, bu toplama MQEnvironment statik değişkenine, sslCertStores 'a ayarlanabilir:

```
MQEnvironment.sslCertStores = crls;
```

Bir bağlantı ayarlandığında, kuyruk yöneticisi tarafından sunulan sertifika aşağıdaki gibi doğrulanır:

1. Kaynak grubundaki ilk CertStore nesnesi, bir CRL sunucusunu tanımlamak için kullanılır. sslCertdepolar tarafından tanımlanır.
2. CRL sunucusuyla bağlantı kurma girişiminde bulunmanız gerekir.
3. Girişim başarılı olursa, sunucu, sertifikan için bir eşleşme için arama yapılır.
   a. Sertifikana geri alınacak bir sertifika bulunursa, arama işlemi sona erir ve bağlantı isteği, MQRC_SSL_CERTIFICATE_FESHRIVE neden kodlarıyla başarısız olur.
   b. Sertifika bulunamazsa, arama işlemi sona ermiş ve bağlantının devam etmesine izin verilir.
4. Sunucuyla iletişim kurma girişimi başarısız olursa, bir sonraki CertStore nesnesi bir CRL sunucusunu tanımlamak için kullanılır ve süreç 2. adımdan yinelenir.

   Bu, derlemdeki son CertStore ise ya da Kaynak Grubu hiçbir CertStore nesnesi içermiyorsa, arama işlemi başarısız oldu ve bağlantı isteği, MQRC_SSL_CERT_STORE_ERROR neden koduyla başarısız olur.

Veri Toplama nesnesi, CertStores ' un kullanıldığı sırayı belirler.

CertStores adlı kaynak grubu da CMQC.SSL_CERT_STORE_PROPERTYkullanılarak da ayarlanabilir. Kolaylık olması nedeniyle, bu özellik, tek bir CertStore ' un bir Kaynak Grubu üyesi olmadan belirtilmesine de olanak tanır.

sslCertStores boş değere ayarlıysa, CRL denetimi gerçekleştirilmez. sslCipherSuite ayarlanmadıysa bu özellik yok sayılır.

Java için WebSphere MQ sınıflarındaki gizli anahtarı yeniden ilişki kurma

Java istemcisi uygulaması için bir WebSphere MQ sınıfları, bir istemci bağlantısında şifreleme için kullanılan gizli anahtarın, gönderilen ve alınan toplam bayt sayısı bakımından yeniden ilişki kurduğunda denetleyebilmesini sağlar.

Uygulama bunu aşağıdaki yollardan biriyle yapabilir: Uygulama bu yollardan birinden fazlasını kullanıyorsa, olağan öncelik kuralları geçerli olur.

- MQEnvironment sınıfındaki sslResetSayı alanını ayarlayarak.
- Bir Hashtable nesnesinde MQC.SSL_RESET_COUNT_PROPERTY ortam özelliği ayarlanarak. Uygulama daha sonra, hashtable 'ı MQEnvironment sınıfındaki `properties` alanına atar ya da hashtable 'ı oluşturucudaki bir MQQueueManager nesnesine geçirir.

The value of the sslResetCount field or environment property MQC.SSL_RESET_COUNT_PROPERTY represents the total number of bytes sent and received by the WebSphere MQ classes for Java client code before the secret key is renegotiated. Gönderilen bayt sayısı, şifrelemeden önceki sayıdır ve alınan bayt sayısı, şifre çözme işleminden sonra gelen sayıdır. Bayt sayısı, Java istemcisi için WebSphere MQ sınıfları tarafından gönderilen ve alınan denetim bilgilerinin de içerilmesine neden olur.

İlk duruma getirme sayısı sıfırsa, varsayılan değer olan gizli anahtar hiçbir zaman yeniden anlaşılamamaktadır. CipherSuite belirtilmediyse, ilk duruma getirme sayısı dikkate alınmaz.

IBM WebSphere MQ classes for Java' ta özelleştirilmiş bir SSLSocketFactory sağlama

Özelleştirilmiş bir JSSE Yuva Üreticisi kullanıyorsanız, MQEnvironment.sslSocketFactory ' yi özelleştirilmiş fabrika nesnesine ayarlayın. Ayrıntılar, farklı JSSE somutlamaları arasında farklılık gösterir.

Farklı JSSE uygulamaları farklı özellikler sağlayabilir. Örneğin, özelleştirilmiş bir JSSE somutlaması, belirli bir şifreleme donanımı modelinin yapılandırılmasına izin verebilir. Buna ek olarak, bazı JSSE sağlayıcıları, anahtar depolarının ve güvenilirlik depolarının programa göre özelleştirilmesine ya da anahtar deposundan değiştirilmek üzere kimlik sertifikası seçmesine izin verir. JSSE ' de, tüm bu uyarlamalar bir üretici sınıfına (javax.net.ssl.SSLSocketFactory) soyutlanır.

Özelleştirilmiş bir SSLSocketFactory uygulamasının nasıl oluşturulabilmesiyle ilgili ayrıntılar için JSSE belgelerinize bakın. Ayrıntılar sağlayıcıdan sağlayıcıya göre değişir, ancak aşağıdaki tipik adımlar dizisi aşağıdaki gibi olabilir:

1. SSLContext üzerinde durağan bir yöntem kullanarak bir SSLContext nesnesi yaratır
2. Uygun KeyManager ve TrustManager uygulamalarıyla (kendi fabrika sınıflarından yaratılan) bu SSLContext ' i başlatın.
3. SSLContext içinden bir SSLSocketFactory yarat

When you have an SSLSocketFactory object, set the MQEnvironment.sslSocketFactory to the customized factory object. Örneğin:

```
javax.net.ssl.SSLSocketFactory sf = sslContext.getSocketFactory();
MQEnvironment.sslSocketFactory = sf;
```

IBM WebSphere MQ classes for Java , IBM WebSphere MQ kuyruk yöneticisine bağlanmak için bu SSLSocketFactory ' yi kullanın. Bu özellik, CMQC.SSL_SOCKET_FACTORY_PROPERTYkullanılarak da ayarlanabilmektedir. sslSocketFactory boş değer olarak ayarlandıysa, JVM ' nin varsayılan SSLSocketFactory değeri kullanılır. sslCipherSuite ayarlanmadıysa bu özellik yok sayılır.

Özel SSLSocketFactories' i kullandığınızda, TCP/IP bağlantı paylaşımının etkisini göz önünde bulundurun. If connection sharing is possible then a new socket is not requested of the SSLSocketFactory supplied, even if the socket produced would be different in some way in the context of a subsequent connection request. Örneğin, sonraki bir bağlantıda farklı bir istemci sertifikası sunulacaksa, bağlantı paylaşımına izin verilmemesi gerekir.

Java için WebSphere MQ sınıflarında JSSE anahtar deposunda ya da güvenilir depoda değişiklik yapılması

JSSE anahtar deposunu ya da güvenilir deposunu değiştirdiğinizde, değişikliklerin yürürlüğe girmesi için bazı işlemleri gerçekleştirmeniz gerekir.

JSSE anahtar deposu ya da güvenilirlik deposunun içeriğini değiştirirseniz ya da anahtar deposunun ya da güvenilir depo dosyasının konumunu değiştirdiğinizde, o sırada çalışan Java uygulamalarına ilişkin WebSphere MQ sınıfları, değişiklikleri otomatik olarak açmaz. Değişikliklerin yürürlüğe girmesi için aşağıdaki işlemler gerçekleştirilmelidir:

- Uygulamalar tüm bağlantılarını kapatmalı ve bağlantı havuzlarındaki kullanılmayan bağlantıları yok etmelidir.
- JSSE sağlayıcınız, bilgileri anahtar deposundan ve güvenilir depodan önbelleğe aldıysa, bu bilgiler yenilenmelidir.

Bu işlemler gerçekleştirildikten sonra, uygulamalar bağlantılarını yeniden yaratabilirler.

Uygulamalarınızı nasıl tasarlamanıza ve JSSE sağlayıcınız tarafından sağlanan işlevlere bağlı olarak, uygulamalarınızı durdurup yeniden başlatmaksızın bu işlemleri gerçekleştirmeniz de mümkün olabilir. Ancak, uygulamaların durdurulması ve yeniden başlatılması en basit çözüm olabilir.

Java için WebSphere MQ sınıflarıyla SSL kullanılırken hata işleme

SSL kullanan bir kuyruk yöneticisine bağlanırken, Java için WebSphere MQ sınıfları tarafından bir dizi neden kodu yayınlanabilir.

Bu bilgiler aşağıdaki listede açıklanmıştır:

MQRC_SSL_NOT_ALLOWED
sslCipherSuite özelliği ayarlıydı, ancak bağ tanımları bağlantısı kullanıldı. SSL ' yi yalnızca istemci bağlantısı destekler.

MQRC_JSSE_ERROR
JSSE sağlayıcısı, WebSphere MQtarafından ele alınmayan bir hata bildirdi. Bunun nedeni, JSSE ile bir yapılandırma sorunu olabilir ya da kuyruk yöneticisi tarafından sunulan sertifikana doğrulanamadı. JSSE tarafından üretilen kural dışı durum, MQException 'daki getCause() yöntemi kullanılarak alınabilir.

MQRC_SSL_INITIALIZATION_ERROR
Belirtilen SSL yapılanış seçenekleri ile bir MQCONN ya da MQCONNX çağrısı yayınlandı, ancak SSL ortamı kullanıma hazırlanırken hata oluştu.

MQRC_SSL_PEER_NAME_MISSMATCH
sslPeeradı özelliğinde belirtilen DN kalıbı, kuyruk yöneticisi tarafından sunulan DN ile eşleşmedi.

MQRC_SSL_PEER_NAME_ERROR
sslPeerAd özelliğinde belirtilen DN kalıbı geçerli değil.

MQRC_UNSUPPORTED_CIPHER_SUITE
sslCipher Suite içindeki CipherSuite takımı JSSE sağlayıcısı tarafından tanınmadı. A full list of CipherSuites supported by the JSSE provider can be obtained by a program using the SSLSocketFactory.getSupportedCipherSuites() method. WebSphere MQ ile iletişim kurmak için kullanılabilecek CipherSuites listesi, “Java için WebSphere MQ sınıflarında SSL CipherSpecs ve CipherSuites” sayfa 678içinde bulunabilir.

MQRC_SSL_CERTIFICATE_FESHEDILDI
Kuyruk yöneticisi tarafından sunulan sertifika, sslCertStores özelliği ile belirtilen bir CRL ' de bulundu. Güvenilir sertifikalar kullanmak için kuyruk yöneticisini güncelleyin.

MQRC_SSL_CERT_STORE_ERROR
Kuyruk yöneticisi tarafından sunulan sertifika için, belirtilen CertStores ' in hiçbiri aranamadı. MQException.getCause() yöntemi, denenen ilk CertStore aranırken oluşan hatayı döndürür. Nedensel kural dışı durum NoSuchElementException, ClassCastKural Dışı Durumu ya da NullPointerÖzel Durumu ise, sslCertMağazalarında belirtilen Derlemin en az bir geçerli CertStore nesnesi içerdiğini doğrulayın.

Java için WebSphere MQ sınıflarında SSL CipherSpecs ve CipherSuites

Bir IBM WebSphere MQ classes for Java uygulamasının kuyruk yöneticisine bağlantı kurabilmesi için, MQI kanalının sunucu ucunda belirtilen CipherSpec 'e ve istemci uçta belirtilen CipherSuite ' e bağlıdır.

For each combination of CipherSpec and CipherSuite, whether a IBM WebSphere MQ classes for Java application can connect to a queue manager depends on the value of the sslFipsRequired field in the MQEnvironment class, or on the value of the environment property CMQC.SSL_FIPS_REQUIRED_PROPERTY.

Bir MQI kanalının sunucu ucunda, CipherSpec adı, DEFINE CHANNEL CHLTYPE (SVRCONN) komutunda SSLCIPH parametresinin değeri olarak belirlenebilir. Bir MQI kanalının istemci ucunda, bir IBM WebSphere MQ classes for Java uygulaması, MQEnvironment sınıfındaki sslCipherSuite alanını ayarlayabilir ya da CMQC.SSL_CIPHER_SUITE_PROPERTYortam özelliğini ayarlayabilir.

Uygulamanızın IBM Java ya da Oracle Java CipherSuite eşlemlerini kullanacak şekilde yapılandırılması

From IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 5, you can configure whether your application uses the default IBM Java CipherSuite to WebSphere MQ CipherSpec mappings, or the Oracle CipherSuite to WebSphere MQ CipherSpec mappings. Bunun için, uygulamanızın bir IBM JRE ya da bir Oracle JRE kullanıyorsa TLS CipherSuites ' i kullanabilirsiniz. The Java System Property `com.ibm.mq.cfg.useIBMCipherMappings` controls which mappings are used. Özellik, aşağıdaki değerlerden biri olabilir:

doğru
IBM Java CipherSuite ' i WebSphere MQ CipherSpec eşlemeleri için kullanın.

Bu değer, varsayılan değerdir.

yanlış
Oracle CipherSuite ' i WebSphere MQ CipherSpec eşlemeleri için kullanın.

Aşağıdaki çizelgede IBM WebSphere MQ tarafından desteklenen CipherSpecs ve eşdeğeri olan CipherSuiteslistesi listelenmektedir. The table also indicates whether a IBM WebSphere MQ classes for Java application can establish a connection to a queue manager if a CipherSpec is specified at the server end of the MQI channel and the equivalent CipherSuite is specified at the client end.

Çizelge 89. CipherSpecs , WebSphere MQ tarafından desteklenir ve eşdeğeri CipherSuites

CipherSpec	**Eşdeğer CipherSuite**	**SFIPS [1] YES değerine ayarlıysa bağlantı yapılabilir mi?**
NULL_MD5	SSL_RSA_WITH_NULL_MD5	Hayır
NULL_SHA	SSL_RSA_WITH_NULL_SHA	Hayır
RC4_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC4_40_MD5 (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
RC4_MD5_US	SSL_RSA_WITH_RC4_128_MD5	Hayır

Çizelge 89. CipherSpecs , WebSphere MQ tarafından desteklenir ve eşdeğeri CipherSuites (devamı var)

CipherSpec	**Eşdeğer CipherSuite**	**SFIPS [1] YES değerine ayarlıysa bağlantı yapılabilir mi?**
RC4_SHA_US	SSL_RSA_WITH_RC4_128_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
RC2_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5 (IBM JRE) SSL_RSA_EXPORT_WITH_RC4_40_MD5 (Oracle JRE)	Hayır
DESTE_SHA_EXPORT	SSL_RSA_WITH_DES_CBC_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
RC4_56_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_RC4_56_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
DES_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
TRIPLE_DES_SHA_US	SSL_RSA_WITH_3DES_EDE_CBC_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır
TLS_RSA_WITH_NULL_SHA256	SSL_RSA_WITH_NULL_SHA256 (IBM JRE) TLS_RSA_WITH_NULL_SHA256 (Oracle JRE)	Hayır [7]
TLS_RSA_WITH_AES_128_CBC_SHA	SSL_RSA_WITH_AES_128_CBC_SHA (IBM JRE) TLS_RSA_WITH_AES_128_CBC_SHA (Oracle JRE)	Evet [5] [7]
TLS_RSA_WITH_AES_128_CBC_SHA256	SSL_RSA_WITH_AES_128_CBC_SHA256 (IBM JRE) TLS_RSA_WITH_AES_128_CBC_SHA256 (Oracle JRE)	Evet [5] [7]
TLS_RSA_WITH_AES_256_CBC_SHA	SSL_RSA_WITH_AES_256_CBC_SHA (IBM JRE) TLS_RSA_WITH_AES_256_CBC_SHA (Oracle JRE)	Evet [5] [7]
TLS_RSA_WITH_AES_256_CBC_SHA256	SSL_RSA_WITH_AES_256_CBC_SHA256 (IBM JRE) TLS_RSA_WITH_AES_256_CBC_SHA256 (Oracle JRE)	Evet [5] [7]
AES_SHA_US[2]		
TLS_RSA_WITH_DES_CBC_SHA[8]	SSL_RSA_WITH_DES_CBC_SHA	Hayır[3]
TLS_RSA_WITH_3DES_EDE_CBC_SHA[8] [9]	SSL_RSA_WITH_3DES_EDE_CBC_SHA	Evet

Çizelge 89. CipherSpecs , WebSphere MQ tarafından desteklenir ve eşdeğeri CipherSuites (devamı var)		
CipherSpec	**Eşdeğer CipherSuite**	**SFIPS [1] YES değerine ayarlıysa bağlantı yapılabilir mi?**
FIPS_WITH_DES_CBC_SHA	SSL_RSA_FIPS_WITH_DES_CBC_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır[4]
FIPS_WITH_3DES_EDE_CBC_SHA	SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA (IBM JRE) Oracle JRE ' ye eşdeğer değildir.	Hayır[6]

Notlar:

1. In a IBM WebSphere MQ classes for Java application, indicate that only FIPS-certified algorithms are to be used by setting the sslFipsRequired field in the MQEnvironment class to `true` and indicate that non-FIPS-certified algorithms can also be used by setting the sslFipsRequired field to `false`. Diğer bir seçenek olarak, CMQC.SSL_FIPS_REQUIRED_PROPERTYortam özelliğini ayarlayın.
2. Bu CipherSpec , eşdeğer bir CipherSuite' e sahip değildir.
3. Bu CipherSpec , 19th Mayıs 2007 tarihinden önce FIPS 140-2 sertifikasıydı.
4. Bu CipherSpec , 19th Mayıs 2007 tarihinden önce FIPS 140-2 sertifikasıydı. FIPS_WIT_DES_CBC_SHA adı geçmiştir ve bu CipherSpec ' in daha önce olduğu (ancak artık değil) FIPS-uyumlu olduğu gerçeğini yansıtır. Bu CipherSpec kullanımdan kaldırıldı ve kullanımı önerilmiyor.
5. Bu CipherSpecs (TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_128_CBC_SHA256, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA256), Gezgin tarafından kullanılan JRE ' ye uygun olmayan uygun olmayan ilke dosyaları uygulanmadığı sürece, WebSphere MQ Gezgini ile bir kuyruk yöneticisine bağlantı sağlamak için kullanılamaz.

   İlke dosyalarına ilişkin ek bilgi için Güvenlik bilgileri başlıklı konuya bakın.
6. The name FIPS_WITH_3DES_EDE_CBC_SHA is historical and reflects the fact that this CipherSpec was previously (but is no longer) FIPS-compliant. Bu CipherSpec kullanımdan kaldırıldı ve kullanımı önerilmiyor.
7. Bu CipherSpecs (TLS_RSA_WITH_NULL_SHA256, TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA256, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA256), IBM JREs 6.0 SR13 FP2 , 7.0 SR4 FP2 ya da sonraki bir sürümü gerektirir.
8. Bu CipherSpecs (TLS_RSA_WITH_3DES_EDE_CBC_SHA, TLS_RSA_WITH_DES_CBC_SHA, TLS_RSA_WITH_RC4_128_SHA256), SSLv3 ya da TLS ' yi kullanabilir. Varsayılan olarak, FIPS etkin olmadığı zaman SSLv3 kullanılır. TLS ' yi kullanmak için **`com.ibm.mq.cfg.preferTLS`** Java System Property özelliğini `true`olarak ayarlayın.
9. Bu CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA kullanımdan kaldırılmıştır. However, it can still be used to transfer up to 32 GB of data before the connection is terminated with error AMQ9288. Bu hatayı önlemek için, üçlü DES kullanmaktan kaçınmanız ya da bu CipherSpeckomutunu kullanırken gizli anahtar sıfırlamayı etkinleştirmeniz gerekir.

İlgili bilgiler

MQI istemcisinde çalıştırma sırasında yalnızca FIPS onaylı CipherSpecs ' in kullanıldığını belirtme

UNIX, Linux ve Windows için Federal Bilgi İşleme Standartları (FIPS)

MQdev web günlüğü: MQ Java, TLS Ciphers, Non-IBM JRE & APAR 'lar: IT06775, IV66840, IT09423, IT10837

MQdev web günlüğü: MQ CipherSpecs ile Java Şifreleme Takımları arasındaki ilişki

Java uygulamaları için WebSphere MQ sınıflarının çalıştırılması

İstemci ya da bağ tanımları kipini kullanarak bir uygulama (main () yöntemi içeren bir sınıf) yazarsanız, Java yorumlayıcını kullanarak programınızı çalıştırın.

Şu komutu kullanın:

```
java -Djava.library.path=library_path MyClass
```

Burada *kitaplık_yolu* , Java kitaplıkları için WebSphere MQ sınıflarının yoludur (bkz. Java kitaplıkları için WebSphere MQ sınıfları).

Java ortamına bağımlı davranış için WebSphere MQ sınıfları

Java için WebSphere MQ sınıfları, farklı WebSphere MQsürümlerine karşı çalışabilecek uygulamalar oluşturmanıza olanak sağlar. Bu konu grubunda, Java sınıflarının bu farklı sürümlere bağımlı davranışı anlatılır.

Java için WebSphere MQ sınıfları, tüm ortamlarda tutarlı bir işlev ve davranış sağlayan bir sınıf çekirdeği sağlar. Bu çekirdeğin dışındaki özellikler, uygulamanın bağlı olduğu kuyruk yöneticisinin yeteneğine bağlıdır.

Burada, burada belirtilenler dışında, sergilenen davranış, kuyruk yöneticisine uygun Uygulama Programlama Başvuruunda açıklanmıştır.

Java için WebSphere MQ sınıflarındaki çekirdek sınıfları

Java içinWebSphere MQ sınıfları, tüm ortamlarda kullanılabilen bir temel sınıf kümesi içerir.

Aşağıdaki sınıf kümesi, temel sınıflar olarak kabul edilir ve tüm ortamlarda yalnızca “Java için WebSphere MQ sınıflarının çekirdek sınıflarına ilişkin kısıtlamalar ve çeşitler” sayfa 682içinde listelenen küçük çeşitlilikler ile kullanılabilir.

- MQEnvironment
- MQException
- MQGetMessageSeçenekleri

  Hariç Tutma:

  - MatchOptions
  - GroupStatus
  - SegmentStatus
  - Bölümleme
- MQManagedObject

  Hariç Tutma:

  - sorgulama ()
  - set ()
- MQMessage

  Hariç Tutma:

  - groupId
  - messageFlags
  - messageSequenceNumarası
  - offset
  - originalLength

- MQPoolServices
- MQPoolServicesOlayı
- MQPoolServicesEventListener
- MQPoolToken
- MQPutMessageSeçenekleri

  Hariç Tutma:

  – knownDestSayısı
  – unknownDestSayısı
  – invalidDestSayı
  – recordFields
- MQProcess
- MQQueue
- MQQueueManager

  Hariç Tutma:

  – begin ()
  – accessDistributionListesi ()
- MQSimpleConnectionYöneticisi
- MQTopic
- MQC

Not:

1. Bazı sabitler çekirdeklere dahil değildir (ayrıntılar için bkz. “Java için WebSphere MQ sınıflarının çekirdek sınıflarına ilişkin kısıtlamalar ve çeşitler” sayfa 682); bunları tamamen taşınabilir programlarda kullanmayın.
2. Bazı platformlar tüm bağlantı kiplerini desteklemez. Bu altyapılarda, yalnızca desteklenen kiplerle ilgili temel sınıfları ve seçenekleri kullanabilirsiniz. (Bkz. “Java için WebSphere MQ sınıflarına ilişkin bağlantı seçenekleri” sayfa 628.)

Java için WebSphere MQ sınıflarının çekirdek sınıflarına ilişkin kısıtlamalar ve çeşitler

Temel sınıflar genellikle tüm ortamlarda tutarlı bir şekilde davranır, aynı MQI çağrıları normal olarak ortam farklılıklarına sahip olsa bile. The behavior is as if a Windows, UNIX or Linux WebSphere MQ queue manager is used, except for the following minor restrictions and variations.

*Java için WebSphere MQ sınıflarındaki MQGMO_ * değerlerine ilişkin kısıtlamalar*

Bazı MQGMO_ * değerleri, tüm kuyruk yöneticileri tarafından desteklenmez.

Aşağıdaki MQGMO_ * değerlerinin kullanılması, bir MQQueue.get() içinden bir MQException yayınına neden olabilir:

MQGMO_SYNCPOINT_IF_PERSISTENT
MQGMO_MARK_SKIP_BACKUT
MQGMO_BROWSE_MSG_UNDER_CURSOR
MQGMO_LOCK
MQGMO_UNLOCK
MQGMO_LOGICAL_ORDER
MQGMO_COMPLETE_MESSAGE
MQGMO_ALL_MSGS_AVALABILIR
MQGMO_ALL_SEGMENTS_AVALABILIR
MQGMO_UNMARKET_BROWSE_MSG
MQGMO_MARK_BROWSE_HANDLE

MQGMO_MARK_BROWSE_CO_OP
MQGMO_UNMARK_BROWSE_HANDLE
MQGMO_UNMARK_BROWSE_CO_OP

Buna ek olarak, Java 'dan kullanıldığında MQGMO_SET_SIGNAL desteklenmez.

*Java için WebSphere MQ sınıflarındaki MQPMRF_ * değerlerine ilişkin kısıtlamalar*
Bunlar yalnızca dağıtım listesine ileti yerleştirilirken kullanılır ve yalnızca dağıtım listelerini destekleyen kuyruk yöneticileri tarafından desteklenmektedir. Örneğin, z/OS kuyruk yöneticileri dağıtım listelerini desteklememektedir.

*Java için WebSphere MQ sınıflarındaki MQPMO_ * değerlerine ilişkin kısıtlamalar*
Bazı MQPMO_ * değerleri, tüm kuyruk yöneticileri tarafından desteklenmiyor

Aşağıdaki MQPMO_ * değerlerinin kullanılması, bir MQQueue.put() ya da MQQueueManager.put () içinden bir MQException yayınına neden olabilir:

MQPMO_LOGICAL_ORDER
MQPMO_NEW_CORREL_ID
MQPMO_NEW_MESSAGE_ID
MQPMO_RESOLVE_LOCAL_Q

*Java için WebSphere MQ sınıflarındaki MQCNO_ * değerlerine ilişkin kısıtlamalar ve varyasyonlar*
Bazı `MQCNO_*` değerleri desteklenmiyor.

- Otomatik istemci yeniden bağlanması, Java için WebSphere MQ sınıfları tarafından desteklenmez. Ayarladığınız `MQCNO_RECONNECT_*` değeri her ne olursa olsun, bağlantı `MQCNO_RECONNECT_DISABLED`' u ayarladığınız gibi hareket etmeye devam eder.
- `MQCNO_FASTPATH` is ignored on queue managers that do not support `MQCNO_FASTPATH`. İstemci bağlantıları tarafından da yoksayılır.

*Java için WebSphere MQ sınıflarındaki MQRO_ * değerlerine ilişkin kısıtlamalar*
Aşağıdaki rapor seçenekleri ayarlanabilir.

MQRO_EXCEPTION_WITH_FULL_DATA
MQRO_EXPIRATION_WITH_FULL_DATA
MQRO_COA_WITHL_FULL_DATA
MQRO_COD_WITH_FULL_DATA
MQRO_DISCARD_MSG
MQRO_PASS_DISCARD_AND_IFADESI

Daha fazla bilgi için bkz. Rapor.

Features outside the core classes of WebSphere MQ classes for Java

Java için WebSphere MQ sınıfları, tüm kuyruk yöneticileri tarafından desteklenmeyen API uzantılarını kullanmak için özel olarak tasarlanmış bazı işlevleri içerir. Bu konuların toplanmasını, kuyruk yöneticisi kullanırken *desteklemeyen* bir kuyruk yöneticisi kullanırken nasıl davrandıkları açıklanmaktadır.

MQQueueManager oluşturucu seçeneğindeki varyasyonlar
MQQueueManager oluşturucularından bazıları isteğe bağlı bir tamsayı bağımsız değişkeni içerir. Bu bağımsız değişkenin bazı değerleri tüm altyapılarda kabul edilmez.

Bir MQQueueManager oluşturucusunun isteğe bağlı bir tamsayı bağımsız değişkeni içerdiğinde, bu, MQI ' ın MQCNO seçenekleri alanına eşlenir ve normal ve hızlı yol bağlantısı arasında geçiş yapmak için kullanılır. Kullanılan tek seçenek MQCNO_STANDARD_BINDING ya da MQCNO_FASTPATH_BINDING ise, bu oluşturucunun genişletilmiş biçimi tüm ortamlarda kabul edilir. Diğer seçenekler, oluşturucunun MQRC_OPTIONS_ERROR ile başarısız olmasına neden olur. Hızlı yol seçeneği CMQC.MQCNO_FASTPATH_BINDING , yalnızca, bunu destekleyen bir kuyruk yöneticisine yönelik bağ tanımları bağlantısıyla tanitilir. Diğer ortamlarda bu değer yoksayılır.

MQQueueManager.begin () yöntemine ilişkin kısıtlamalar
Bu yöntem, bağ tanımları kipindeki UNIX, Linuxya da Windows sistemlerinde yalnızca WebSphere MQ kuyruk yöneticisine karşı kullanılabilir. Tersi durumda, MQRC_ENVIRONMENT_ERROR ile başarısız olur.

Daha ayrıntılı bilgi için bkz. "Java için WebSphere MQ sınıfları kullanılarak JTA/JDBC eşgüdümü" sayfa 670 .

MQGetMessageSeçenekleri alanlarındaki çeşitlemeler
Bazı kuyruk yöneticileri Sürüm 2 MQGMO yapısını desteklemez, bu nedenle bazı alanları varsayılan değerlerine ayarlamalısınız.

Sürüm 2 MQGMO yapısını desteklemeyen bir kuyruk yöneticisi kullanırken, aşağıdaki alanları varsayılan değerlerine ayarlı olarak bırakın:

GroupStatus
SegmentStatus
Bölümleme

Ayrıca, MatchOptions alanı yalnızca MQMO_MATCH_MSG_ID ve MQMO_MATCH_COREL_ID ' yi destekler. Bu alanlara desteklenmeyen değerler koyarsanız, sonraki MQDestination.get() işlemi MQRC_GMO_ERROR ile başarısız olur. Kuyruk yöneticisi Sürüm 2 MQGMO yapısını desteklemiyorsa, başarılı bir MQDestination.get() işleminden sonra bu alanlar güncellenmez.

Java için WebSphere MQ sınıflarındaki dağıtım listelerindeki kısıtlamalar
Tüm kuyruk yöneticileri bir MQDistributionListolanağını açmanıza izin vermez.

Dağıtım listeleri yaratmak için aşağıdaki sınıflar kullanılır:

MQDistributionList
MQDistributionListÖğesi
MQMessageTracker

You can create and populate MQDistributionLists and MQDistributionListItems in any environment, but not all queue managers allow you to open an MQDistributionList. Özellikle, z/OS kuyruk yöneticileri dağıtım listelerini desteklemez. MQRC_OD_ERROR içinde bu tür bir kuyruk yöneticisi sonucu kullanılırken bir MQDistributionList açılmaya çalışılıyor.

MQPutMessageSeçenekleri alanlarındaki çeşitlemeler
Bir kuyruk yöneticisi dağıtım listelerini desteklemiyorsa, bazı MQPMO alanları farklı davranılır.

MQPMO ' daki dört alan, MQPutMessageSeçenekleri sınıfında aşağıdaki üye değişkenleri olarak görsel olarak gerçekleştirilir:

knownDestSayısı
unknownDestSayısı
invalidDestSayı
recordFields

Bu alanlar öncelikli olarak dağıtım listeleriyle kullanılmak üzere tasarlanmıştır. Ancak, dağıtım listelerini destekleyen bir kuyruk yöneticisi, bir MQPUT ' dan sonra tek bir kuyruğa gelen DestCount alanlarını da doldurur. Örneğin, kuyruk bir yerel kuyruğa çözülürse, knownDestSayı 1 olarak ayarlanır ve diğer iki sayı alanı 0 olarak ayarlanır.

Kuyruk yöneticisi dağıtım listelerini desteklemiyorsa, bu değerler aşağıdaki gibi benzetimli olur:

- Put () başarılı olursa, unknownDestSayısı 1 olarak ayarlanır ve diğerleri 0 değerine ayarlanır.
- Put () başarısız olursa, invalidDestSayı 1 olarak ayarlanır ve diğerleri 0 değerine ayarlanır.

Dağıtım listeleriyle birlikte recordFields değişkeni kullanılır. Ortam ne olursa olsun, herhangi bir zamanda recordFields içine bir değer yazılabilir. MQPutMessageOptions nesnesi, MQDistributionList.put () yerine sonraki bir MQDestination.put() ya da MQQueueManager.put () üzerinde kullanılırsa yoksayılır.

Java için WebSphere MQ sınıflarına sahip MQMD alanlarındaki kısıtlamalar
İleti bölümlemesiyle ilgili bazı MQMD alanları, segmentasyonu desteklemeyen bir kuyruk yöneticisi kullanılırken varsayılan değerlerinde bırakılmalıdır.

Aşağıdaki MQMD alanları büyük oranda ileti bölümlemesiyle ilgilenir:

GroupId
MsgSeqNumarası
Göreli Konum
MsgFlags
OriginalLength

Bir uygulama bu MQMD alanlarından herhangi birini varsayılan değerlerine göre ayarlarsa ve bu alanları desteklemeyen bir kuyruk yöneticisine bir put () ya da get () işlemi yaparsa, put () ya da get (), MQRC_MD_ERROR ile MQException ortaya çıkar. Böyle bir kuyruk yöneticisiyle başarılı bir put () ya da get (), her zaman MQMD alanlarını varsayılan değerlerine ayarlanmış olarak bırakır. İleti gruplamasını ve kesimlere ayırma özelliğini desteklemeyen bir kuyruk yöneticisine karşı çalışan bir Java uygulamasına gruplandırılmış ya da bölümlenmiş bir ileti göndermeyin.

Bir Java uygulaması, bu alanları desteklemeyen bir kuyruk yöneticisinden ileti almaya çalışırsa ve alınacak fiziksel ileti bölümlenmiş bir ileti grubunun bir parçasıysa (yani, MQMD alanları için varsayılan olmayan değerlere sahip), hata olmadan alınır. Ancak, MQMessage 'daki MQMD alanları güncellenmez; MQMessage biçimi özelliği MQFMT_MD_EXTENSION değerine ayarlıdır ve gerçek ileti verilerinin başında, yeni alanların değerlerini içeren bir MQMDE yapısı bulunur.

CICS Transaction Server altında Java için WebSphere MQ sınıflarına ilişkin kısıtlamalar
CICS Transaction Server for z/OS ortamında, yalnızca ana (birinci) iş parçacığın CICS ya da WebSphere MQ çağrılarını yayınlamaya izin verilir.

Bu WebSphere MQ JMS sınıflarının bir CICS Java uygulamasında kullanım için desteklenmediğini unutmayın.

Bu nedenle, bu ortamdaki iş parçacıkları arasında MQQueueManager ya da MQQueue nesnelerini paylaşmak ya da alt iş parçacığının üzerinde yeni bir MQQueueManager yaratmak olanaklı değildir.

Java platformu Enterprise Editioniçindeki Java uygulamaları için IBM WebSphere MQ sınıfları çalıştırılıyor
There are certain restrictions and design considerations that must be taken into account before using IBM WebSphere MQ classes for Java in Java EE

Java içinIBM WebSphere MQ sınıfları, bir Java EE ortamında kullanıldığında kısıtlamalar içerir. Bir Java EE ortamı içinde çalışan Java uygulaması için bir IBM WebSphere MQ sınıfı tasarlarken, uygularken ve yönetirken dikkate alınması gereken ek konular da vardır. Bu kısıtlamalar ve dikkat edilmesi gereken noktalar aşağıdaki bölümlerde açıklanmaktadır.

JTA hareketleri kısıtlamaları

Java için IBM WebSphere MQ sınıflarını kullanan uygulamalar için desteklenen tek hareket yöneticisi IBM WebSphere MQ kendisidir. JTA denetimi altındaki bir uygulama, Javaiçin IBM WebSphere MQ sınıflarını kullanabilse de, bu sınıflar aracılığıyla gerçekleştirilen tüm işler JTA iş birimleri tarafından denetlenmez. Bunun yerine, JTA arabirimleri aracılığıyla uygulama sunucusu tarafından yönetilenlerden ayrı olarak yerel iş birimleri oluştururlar. Özellikle, JTA hareketinin herhangi bir geriye işlenmesi gönderilen ya da alınan iletilerin geri alınmasına neden olmaz. Bu kısıtlama, uygulama ya da bean tarafından yönetilen hareketler ve taşıyıcı tarafından yönetilen hareketler ve tüm Java EE taşıyıcıları için geçerlidir. İleti alışverişi işlemini, uygulama sunucusu eşgüdümlü hareketler içinde IBM WebSphere MQ ile doğrudan gerçekleştirmek için, bunun yerine JMS için IBM WebSphere MQ sınıfları kullanılmalıdır.

İş parçacığı yaratma

Java içinIBM WebSphere MQ sınıfları, çeşitli işlemler için iç iş parçacıkları yaratır. Örneğin, bir yerel kuyruk yöneticisinde doğrudan aramak için BAĞLAMALAR kipinde çalışırken, çağrılar Javaiçin IBM WebSphere MQ sınıfları tarafından dahili olarak yaratılan bir 'işçi' iş parçacığında yapılır. Örneğin, bir bağlantı havuzundan kullanılmayan bağlantıları temizlemek ya da sonlandırılmış yayınlama/abone olma uygulamalarına ilişkin abonelikleri kaldırmak için diğer iş parçacıkları içeride yaratılabilir.

Bazı Java EE uygulamaları (örneğin, EJB ve Web kapsayıcılarında çalıştırılanlar) yeni iş parçacıkları yaratmamalıdır. Bunun yerine, uygulama sunucusu tarafından yönetilen ana uygulama iş parçacıklarına ilişkin tüm işler gerçekleştirilmelidir. Uygulamalar Javaiçin IBM WebSphere MQ sınıflarını kullanırken, uygulama sunucusu uygulama kodunu ve Java kodu için IBM WebSphere MQ sınıflarını ayırt edemeyebilir; bu nedenle, daha önce tanımlanan iş parçacıkları uygulamanın kapsayıcı belirtimiyle uyumlu olmasını sağlar. JMS içinIBM WebSphere MQ sınıfları bu Java EE belirtimlerini bozmaz ve bunun yerine kullanılabilir.

Güvenlik kısıtlamaları

Security policies implemented by an application server might prevent certain operations that are undertaken by the IBM WebSphere MQ classes for Java API, such as creating and operating new threads of control (as described in the preceding sections).

Örneğin, uygulama sunucuları genellikle varsayılan olarak Java Security tarafından devre dışı bırakılır ve uygulama sunucusuna özgü bazı yapılandırma yoluyla etkinleştirilmesine izin verir (bazı uygulama sunucuları aynı zamanda Java Security içinde kullanılan ilkelerin daha ayrıntılı bir şekilde yapılandırılmasına olanak tanır). Java Security etkinleştirildiğinde, Java için IBM WebSphere MQ sınıfları, uygulama sunucusu için tanımlanan Java Güvenlik ilkesi okuma kurallarını bozabilir ve API, işlev yapmak için gereksinim duyduğu tüm iş parçacıklarını oluşturamayabilir. To prevent problems with thread management, the use of IBM WebSphere MQ classes for Java is not supported in environments where Java Security is enabled.

Uygulama yalıtma konuları

Uygulamaların Java EE ortamında çalıştırılması amaçlanan bir avantaj uygulamasıdır. The design and implementation of IBM WebSphere MQ classes for Java predate the Java EE environment. IBM WebSphere MQ Java için sınıflar, uygulama yalıtma kavramını desteklemeyen bir şekilde kullanılabilir. Bu alandaki dikkat edilecek noktalara ilişkin örnekler arasında şunlar yer alır:

- MQEnvironment sınıfındaki statik (JVM process-wide) ayarlarının kullanımı; örneğin:
  - bağlantı kimliği ve kimlik doğrulaması için kullanılacak kullanıcı kimliği ve parola
  - istemci bağlantıları için kullanılan anasistem adı, kapı ve kanal
  - Güvenli istemci bağlantıları için SSL yapılandırması

  Bir uygulamanın yararına ilişkin MQEnvironment özelliklerinin değiştirilmesi, aynı özellikleri kullanan diğer uygulamaları da etkiler. Java EEgibi çok sayıda uygulama ortamında çalışırken, her bir uygulamanın, süreç genelindeki MQEnvironment sınıfında yapılandırılan özellikleri varsayılan olarak kullanmak yerine, belirli bir özellik kümesiyle MQQueueManager nesnelerini yaratma yoluyla kendi ayrı yapılandırmasını kullanması gerekir.
- MQEnvironment sınıfı, aynı JVM süreci içinde Java için IBM WebSphere MQ sınıflarını kullanan tüm uygulamalarda genel olarak hareket eden bir dizi durağan yöntem sunar ve bu davranışı belirli uygulamalar için geçersiz kılmanın bir yolu yoktur. Örnekler:
  - SSL özelliklerini yapılandırma; örneğin, anahtar deposunun yeri
  - istemci kanal çıkışlarını yapılandırma
  - tanılama izlemesini geçerli ya da geçersiz kılma
  - Kuyruk yöneticilerine yönelik bağlantıların kullanımını eniyilemek için kullanılan varsayılan bağlantı havuzunu yönetme

Bu yöntemlerin çağrılması, aynı Java EE ortamında çalışan tüm uygulamaları etkiler.

- Bağlantı havuzlama, aynı kuyruk yöneticisine birden çok bağlantı oluşturma işlemini eniyilemek için etkinleştirilir. Varsayılan bağlantı havuzu yöneticisi, süreç genelinde ve birden çok uygulama tarafından paylaşılır. Changes to connection pool configuration, such as replacing the default connection manager for one application using the MQEnvironment.setDefaultConnectionManager() method therefore affects other applications running in the same Java EE application server.
- SSL, MQEnvironment sınıfını ve MQQueueManager nesne özelliklerini kullanarak Java için IBM WebSphere MQ sınıflarını kullanan uygulamalar için yapılandırılır. Uygulama sunucusunun yönetilen güvenlik yapılandırmasıyla bütünleştirilmedi. You must ensure that you configure IBM WebSphere MQ classes for Java appropriately to provide your required level of security, and not use the application server configuration.

Bağ tanımları kipi kısıtlamaları

IBM WebSphere MQ and WebSphere Application Server can be installed on the same machine such that the major versions of the queue manager and of the IBM WebSphere MQ resource adapter (RA) shipped in WebSphere Application Server are different. For instance WebSphere Application Server Version 7.0, which ships an IBM WebSphere MQ RA level of 7.0.1, can be installed on the same machine as a Version 6.0 queue manager.

Kuyruk yöneticisi ve kaynak bağdaştırıcısı ana sürümleri farklıysa, bağ tanımları bağlantıları kullanılamaz. Kaynak bağdaştırıcısı kullanılarak kuyruk yöneticisine WebSphere Application Server ile olan bağlantılar istemci tipi bağlantılarını kullanmalıdır. Sürümler aynıysa, bağ tanımları bağlantıları kullanılabilir.

JMS için WebSphere MQ sınıflarının kullanılması

Java Message Service için WebSphere MQ sınıfları (JMS için WebSphere MQ sınıfları), WebSphere MQile birlikte verilen JMS sağlayıcısıdır. javax.jms paketinde tanımlanan arabirimlerin yanı sıra, JMS için WebSphere MQ sınıfları JMS API ' ya iki uzantı kümesi sağlar.

JMS belirtimi, uygulamaların ileti alışverişi işlemlerini gerçekleştirmek için kullanabilecekleri bir arabirim kümesini tanımlar. javax.jms paketi JMS arabirimlerini tanımlar ve bir JMS sağlayıcısı, belirli bir ileti alışverişi ürünü için bu arabirimleri uygular. WebSphere MQ sürüm 7.5 şu anda JMS 1.1 belirtimini kullanır. JMS için WebSphere MQ sınıfları, WebSphere MQiçin JMS arabirimlerini uygulayan bir JMS sağlayıcısıdır.

JMS belirtimi, ConnectionFactory ve Hedef nesnelerin denetlenmesini bekler. Bir denetimci, denetlenen nesneleri merkezi bir havuzda yaratır ve bakımını yapar; JMS uygulaması bu nesneleri Java Naming and Directory Interface (JNDI) kullanarak alır. WebSphere MQ için JMS sınıfları denetimli nesne kullanımını destekler ve denetimci, denetlenen nesneleri yaratmak ve bakımını yapmak için WebSphere MQ JMS denetim aracını ya da WebSphere MQ Explorer olanağını kullanabilir.

JMS için WebSphere MQ sınıfları da JMS API ' ya iki uzantı kümesi sağlar. Bu uzantıların ana odağı, yürütme sırasında bağlantı üreticilerinin ve hedeflerin dinamik olarak oluşturulması ve yapılandırılması ile ilgilidir; ancak, uzantılar sorunun saptanması için işlev gibi ileti sistemiyle doğrudan ilgili olmayan bir işlev de sağlar.

WebSphere MQ JMS uzantıları
JMS için önceki WebSphere MQ sınıflarının önceki yayın düzeyleri, MQConnectionFactory, MQQueue ve MQTopic nesneleri gibi nesnelerde uygulanan uzantılar içerir. Bu nesneler, WebSphere MQ' e özgü özellikleri ve yöntemlere sahiptir. Nesneler yönetilebilir ya da bir uygulama, çalıştırma zamanında nesneleri dinamik olarak yaratabilir. JMS için WebSphere MQ sınıflarının bu yayın düzeyi, artık WebSphere MQ JMS uzantıları olarak bilinen bu uzantıların bakımını yapar. Bu uzantıları kullanan uygulamalar, değişiklik yapılmaksızın, kullanmaya devam edebilirsiniz.

IBM JMS uzantıları
JMS için WebSphere MQ sınıflarının bu yayın düzeyi, ileti sistemi sistemi olarak WebSphere MQ 'a özgü olmayan, JMS API' ya daha soysal bir uzantı kümesi sağlar. Bu uzantılar, IBM JMS uzantıları olarak bilinir ve aşağıdaki geniş hedeflere sahiptir:

- IBM JMS sağlayıcıları arasında daha yüksek bir tutarlılık düzeyi sağlamak için

- İki IBM ileti sistemi sistemi arasında bir köprü uygulaması yazmanın daha kolay olmasını sağlamak için
- Bir uygulamanın bir IBM JMS sağlayıcısından başka bir uygulamayı daha kolay kapmasını sağlamak için

Uzantılar, Message Service Client for C/C++ ve Message Service Client for .NETiçinde sağlanan işlevlere benzer bir işlev sağlar.

JMS için neden WebSphere MQ sınıflarını kullanmalıyım?

JMS için WebSphere MQ sınıflarının kullanılması aşağıdaki avantajlara sahiptir:

- JMS becerilerini yeniden kullanabilirsiniz.

  JMS için WebSphere MQ sınıfları, ileti sistemi sistemi olarak WebSphere MQ için JMS arabirimlerini uygulayan bir JMS sağlayıcısıdır. If your organization is new to WebSphere MQ, but already has JMS application development skills, you might find it easier to use the familiar JMS API to access WebSphere MQ resources rather than one of the other APIs provided with WebSphere MQ.
- JMS, Java Platform, Enterprise Edition ürününün ayrılmaz bir parçasıdır (Java EE).

  JMS is the natural API to use for messaging on the Java EE platform. Java EE ile uyumlu her uygulama sunucusunun bir JMS sağlayıcısı içermesi gerekir. JMS 'yi uygulama istemcilerinde, sunucu uygulamacıklarında, JavaServer sayfalarında (JSP' ler), kurumsal Java çekirdeklerini (EJB ' leri) ve ileti odaklı Bean 'leri (MDBs) kullanabilirsiniz. Note in particular that Java EE applications use MDBs to process messages asynchronously, and all messages are delivered to MDBs as JMS messages.
- Bir yönetici, merkezi bir havuzda JMS denetimli nesneleri yaratabilir ve bakımını yapabilir ve JMS uygulamaları için WebSphere MQ sınıfları Java Naming and Directory Interface (JNDI) kullanarak bu nesneleri alabilir.

  JMS bağlantı üreticileri ve hedefleri, kuyruk yöneticisi adları, kanal adları, bağlantı seçenekleri, kuyruk adları ve konu adları gibi WebSphere MQ ' ya özgü bilgileri içerir. Bağlantı üreticileri ve hedefleri, yönetilen nesneler olarak depolanırsa, bu bilgiler bir uygulamaya değişmez olarak kodlanmaz. Bu nedenle bu düzenleme, temeldeki WebSphere MQ yapılanışından bir derece bağımsızlığa sahip bir uygulama sağlar.
- JMS, uygulama taşınabilirliği sağlayabilen bir endüstri standardı API 'sidir.

  Bir JMS uygulaması, yönetilen nesneler olarak saklanan bağlantı fabrikalarını ve hedeflerini almak için JNDI kullanabilir ve ileti alışverişi işlemlerini gerçekleştirmek için yalnızca javax.jms paketinde tanımlı olan arabirimleri kullanır. Uygulama daha sonra JMS için WebSphere MQ sınıfları gibi herhangi bir JMS sağlayıcısından tamamen bağımsızdır ve uygulamada herhangi bir değişiklik yapılmadan bir JMS sağlayıcısından başka bir JMS sağlayıcısından başka bir JMS sağlayıcısından dışa aktarılabilir.

  Belirli bir uygulama ortamında JNDI kullanılabilir değilse, JMS uygulaması için bir WebSphere MQ sınıfları, yürütme sırasında devingen olarak bağlantı üreticileri ve hedefler yaratmak ve yapılandırmak için JMS API uzantılarını kullanabilir. Uygulama daha sonra tamamen kendi kendine bulundurur, ancak JMS sağlayıcısı olarak JMS için WebSphere MQ sınıflarıyla bağlantılıdır.
- Köprü uygulamaları JMS kullanarak yazmanız daha kolay olabilir.

  Bir köprü uygulaması, bir ileti sistemi sisteminden ileti alan ve bunları başka bir ileti sistemi sistemine gönderen bir uygulamadır. Bir köprü uygulaması yazmak, ürüne özgü API ' ler ve ileti biçimleri kullanılarak karmaşık olabilir. Bunun yerine, her ileti sistemi için bir tane olmak üzere iki JMS sağlayıcıyı kullanarak bir köprü uygulaması yazabilirsiniz. Uygulama daha sonra yalnızca bir API, JMS API ' yı kullanır ve yalnızca JMS iletilerini işler.

JMS için WebSphere MQ sınıflarıyla çalışmaya başlama

Bu konu, JMS için WebSphere MQ sınıflarına genel bir bakış sağlar ve JMS için WebSphere MQ sınıflarını kullanmadan önce bilmeniz gerekenleri gösterir.

JMS için WebSphere MQ sınıflarına ilişkin önkoşullar

JMS uygulamaları için WebSphere MQ sınıflarını geliştirmek ve çalıştırmak için, bazı yazılım bileşenlerine önkoşul olarak gereksinim duyarsınız.

JMS için WebSphere MQ sınıflarına ilişkin önkoşullarla ilgili en son bilgiler için WebSphere MQ benioku dosyasına bakın.

JMS uygulamaları için WebSphere MQ sınıfları geliştirmek için bir Java 2 Yazılım Geliştirme Takımı 'na (SDK) gerek vardır. İşletim sisteminizle desteklenen JDKS ' lerin ayrıntıları WebSphere MQ Sistem gereksinimleri sayfasında bulunabilir. Bkz. WebSphere MQ Gereksinimleri.

JMS uygulamaları için WebSphere MQ sınıflarını çalıştırmak için aşağıdaki yazılım bileşenlerine gereksinim duyarsınız:

- Bir WebSphere MQ kuyruk yöneticisi
- Uygulamaları çalıştırdığınız her sistem için bir Java Runtime Environment (JRE)

FIPS 140-2 sertifikalı şifreleme modülleri kullanmak için SSL bağlantısına gereksinim duyuyorsanız, IBM Java JSSE FIPS sağlayıcısına (IBMJSSEFIPS) gerek duyarsınız. Sürüm 5 ya da sonraki sürümlerde her IBM Java 2 SDK ve JRE, IBMJSSEFIPS ' ler içerir.

You can use Internet Protocol Version 6 (IPv6) addresses in your WebSphere MQ classes for JMS applications provided IPv6 addresses are supported by your Java virtual machine (JVM) and the TCP/IP implementation on your operating system. WebSphere MQ JMS yönetim aracı (bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896), IPv6 adreslerini de kabul eder.

WebSphere MQ JMS yönetim aracı ve WebSphere MQ Explorer, yönetilen nesneleri depolayan bir dizin hizmetine erişmek için JNDI (Java Naming and Directory Interface; Java Adlandırma ve Dizin Arabirimi) olanağını kullanır. JMS uygulamaları için WebSphere MQ sınıfları, bir dizin hizmetinden yönetilen nesneleri almak için de JNDI kullanabilir. Hizmet sağlayıcısı, JNDI çağrılarını dizin hizmetine eşleyerek bir dizin hizmetine erişim sağlayan bir koddur. Aşağıdaki hizmet sağlayıcılar, JMS için WebSphere MQ sınıflarıyla birlikte sağlanır:

- ldap.jar ve providerutil.jarkütüklerinde bir LDAP (Lightweight Directory Access Protocol) hizmet sağlayıcısı. LDAP hizmet sağlayıcısı, LDAP sunucusuna dayalı bir dizin hizmetine erişim sağlar.
- A file system service provider in the files fscontext.jar and providerutil.jar. Dosya sistemi hizmet sağlayıcısı, yerel dosya sistemine dayalı olarak bir dizin hizmetine erişim sağlar.

LDAP sunucusuna dayalı bir dizin hizmeti kullanmak istiyorsanız, bir LDAP sunucusu kurmalı ve yapılandırmalı ya da var olan bir LDAP sunucusuna erişiminiz olmalıdır. Özellikle, Java nesnelerini saklamak için LDAP sunucusunu yapılandırmanız gerekir. LDAP sunucunuzu kurmaya ve yapılandırmaya ilişkin bilgi için sunucunuzla birlikte sağlanan belgelere bakın.

Preparing JMS programs for the IBM WebSphere MQ client for HP Integrity NonStop Server

This topic explains what you need to know before you develop and run JMS programs for the IBM WebSphere MQ client for HP Integrity NonStop Server.

JMS için IBM WebSphere MQ sınıfları, HP Integrity NonStop Server kuruluşu için IBM WebSphere MQ istemcisinin bir parçası olarak kurulur. Kuruluşun içeriğinin bir özetine ilişkin ayrıntılar için Dosya sistemibaşlıklı konuya bakın.

İstemci işlevlerinin bazı yönleri anasistem işletim sistemine özgütür. HP Integrity NonStop Serveriçin IBM WebSphere MQ istemcisi için desteklenen özellikler hakkında daha fazla bilgi için bkz. IBM WebSphere MQ Client for HP Integrity NonStop Server desteklenen ortamlar ve özellikler.

Önkoşullar

JMS uygulamalarını oluşturmak ve çalıştırmak için, *HP Integrity NonStop Server for Java* bileşeni kurulu ve kullanılabilir olmalıdır.

Ayarla

JMS için IBM WebSphere MQ sınıflarını kullanabileceğiniz uygulamaların çalıştırılması ve oluşturulması için ortamın ayarlanmasıyla ilgili bilgi edinmek için bkz. “JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri” sayfa 694.

Bir kuyruk yöneticisini istemci uygulamalarından gelen bağlantıları kabul edecek şekilde yapılandırmak için gereken adımlarla ilgili bilgi için bkz. “JMS uygulamaları için WebSphere MQ sınıflarına ilişkin kuruluş sonrası ayarları” sayfa 740.

JMS ortamına ilişkin IBM WebSphere MQ sınıflarınızın doğrulanmasına ilişkin bilgi edinmek için “JMS için WebSphere MQ sınıfları için noktadan noktaya kuruluş doğrulama sınaması” sayfa 743konusuna bakın.

Uygulamalar yazılıyor

JMS uygulamalarını yazma hakkında daha fazla bilgi için bkz. “JMS uygulamaları için WebSphere MQ sınıflarının yazılması” sayfa 773.

IBM WebSphere MQ JMS yönetim aracını kullanma hakkında daha fazla bilgi için bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896.

Örnekler

Kuruluşun aşağıdaki alt dizininde örnek uygulamalar sağlanır: `opt/mqm/samp/jms`.

Örnekleri çalıştırmak için gereken yapılandırma adımlarına ilişkin daha fazla bilgi için bkz. “Örnek programların hazırlanması ve çalıştırılması” sayfa 104.

Sorun çözme

Sorunların çözülmesine ilişkin bilgi için bkz. “JMS için IBM WebSphere MQ sınıflarıyla ilgili sorunların çözülmesi” sayfa 764.

JMS için WebSphere MQ sınıflarının kurulması ve yapılandırılması

Bu kısımda, JMS için WebSphere MQ sınıflarını kurduğunuzda yaratılan ve WebSphere MQ sınıflarının kuruluştan sonra JMS için nasıl yapılandırılacağı anlatıldığında yaratılan dizinler ve dosyalar açıklanmaktadır.

İlgili kavramlar

“JMS için IBM WebSphere MQ sınıfları için kurulu olan” sayfa 691
JMS için IBM WebSphere MQ sınıfları kurduğunuzda, bir dizi dosya ve dizin oluşturulur. On Pencereler some configuration is performed during installation by automatically setting environment variables. Diğer altyapılarda ve bazı Windows ortamlarında, JMS uygulamaları için IBM WebSphere MQ sınıflarını çalıştırabilmeniz için ortam değişkenlerini ayarlamalısınız.

“Java güvenlik yöneticisi altındaki JMS uygulamaları için WebSphere MQ sınıflarının çalıştırılması” sayfa 700
WebSphere MQ classes for JMS can run with the Java security manager enabled. Güvenlik yöneticisini etkinleştiren uygulamaları başarıyla çalıştırmak için, Java sanal makinenizi (JVM) uygun bir ilke yapılandırma dosyasıyla yapılandırmalısınız.

“IBM WebSphere MQ kaynak bağdaştırıcısı” sayfa 704
Kaynak bağdaştırıcısı, uygulama sunucusunda çalışan uygulamaların IBM WebSphere MQ kaynaklarına erişmelerini sağlar. Gelen ve giden iletişimi destekler.

“JMS uygulamaları için WebSphere MQ sınıflarına ilişkin kuruluş sonrası ayarları” sayfa 740
This topic tells you what authorities WebSphere MQ classes for JMS applications need in order to access the resources of a queue manager. Ayrıca, bağlantı kipleri tanıtılır ve uygulamaların istemci kipinde bağlanabilmesi için kuyruk yöneticisinin nasıl yapılandırılacağı anlatılır.

“JMS için WebSphere MQ sınıfları için noktadan noktaya kuruluş doğrulama sınaması” sayfa 743

JMS için WebSphere MQ sınıflarıyla birlikte bir noktadan noktaya kuruluş doğrulama sınaması (IVT) programı sağlanır. Program, bağ tanımlarında ya da istemci kipinde bir kuyruk yöneticisine bağlanır ve kuyruğa SYSTEM.DEFAULT.LOCAL.QUEUEve daha sonra, iletiyi kuyruktan alır. Program, yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratabilir ve yapılandırabilir ya da bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanabilir.

“JMS için WebSphere MQ sınıfları için yayınlama/abone olma kuruluş doğrulama sınaması” sayfa 746
JMS için WebSphere MQ sınıflarıyla birlikte bir yayınlama/abone olma kuruluş doğrulama testi (IVT) programı sağlanır. Program, bağ tanımları ya da istemci kipinde bir kuyruk yöneticisine bağlanır, bir konuya abone olur, konuyla ilgili bir ileti yayınlar ve daha sonra, yeni yayınladığı iletiyi alır. Program, yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratabilir ve yapılandırabilir ya da bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanabilir.

“ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750
IVT programı bir EAR dosyası olarak sağlanır. Programı kullanmak için, programı konuşlandırmanız ve JCA kaynakları olarak bazı nesneler tanımlamanız gerekir.

“Kaynak bağdaştırıcısının giden iletişim için yapılandırılması” sayfa 721
Giden iletişim yapılandırmak için bir ConnectionFactory nesnesinin özelliklerini ve denetlenen bir hedef nesneyi tanımlayın.

“OSGi desteği” sayfa 763
OSGi, uygulamaların kod paketleri olarak konuşlandırılmasını destekleyen bir çerçeve sağlar. Dokuz OSGi kod paketi IBM WebSphere MQ classes for JMS ' nin bir parçası olarak sağlanır.

“JMS için IBM WebSphere MQ sınıflarıyla ilgili sorunların çözülmesi” sayfa 764
Kuruluş doğrulama programlarını çalıştırabilir ve izleme ve günlük olanaklarını kullanarak sorunları araştırabilirsiniz.

İlgili görevler

“WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması” sayfa 753
IBM WebSphere MQ kaynak bağdaştırıcısının kurulması ve WebSphere Application Server CE ' de kuruluş doğrulama sınaması (IVT) uygulaması çalıştırılması.

“Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması” sayfa 755
If you want to use a different queue, queue manager, port, host, channel, or use bindings mode instead of client mode, you must modify the IVT application and associated scripts in WebSphere Application Server CE before deploying the resource adapter or IVT application.

“Deploying the IVT application in JBoss with a custom IBM WebSphere MQ environment” sayfa 758
IBM WebSphere MQ kaynak bağdaştırıcısını JBossiçine kurarken, istemci kipi yerine farklı bir kuyruk, kuyruk yöneticisi, kapı, anasistem, kanal ya da bağ tanımları kipini kullanmak istiyorsanız, kaynak bağdaştırıcısı ya da IVT uygulamasını konuşlandırmadan önce JBoss ' ta IVT uygulamasını ve ilişkili komut dosyalarını değiştirmelisiniz.

IBM WebSphere MQ kaynak bağdaştırıcısı için sorun belirleme

İlgili başvurular

“JMS için WebSphere MQ sınıflarıyla sağlanan komut dosyaları” sayfa 762
JMS için WebSphere MQ sınıfları kullanılırken gerçekleştirilmesi gereken genel görevlere yardımcı olmak için bir dizi komut dosyası sağlanmıştır.

JMS için IBM WebSphere MQ sınıfları için kurulu olan

JMS için IBM WebSphere MQ sınıfları kurduğunuzda, bir dizi dosya ve dizin oluşturulur. On Pencereler some configuration is performed during installation by automatically setting environment variables. Diğer altyapılarda ve bazı Windows ortamlarında, JMS uygulamaları için IBM WebSphere MQ sınıflarını çalıştırabilmeniz için ortam değişkenlerini ayarlamalısınız.

Çoğu işletim sistemi için,JMS' in IBM WebSphere MQ sınıfları, IBM WebSphere MQürününü kurduğunuzda isteğe bağlı bir bileşen olarak kurulur. HP Integrity NonStop Serveriçin IBM WebSphere MQ istemcisi için, varsayılan olarak JMS ' nin IBM WebSphere MQ sınıfları kurulur. IBM WebSphere MQürününü kurmaya ilişkin ek bilgi için aşağıdaki başlara bakın:

WebSphere MQ sunucusunun kurulması

IBM WebSphere MQ istemcisi kurulması

Çizelge 90 sayfa 692 , her altyapıda JMS dosyalarına ilişkin IBM WebSphere MQ sınıflarının kurulu olduğu yeri gösterir.

Çizelge 90. JMS kuruluş dizinlerine ilişkinIBM WebSphere MQ sınıfları	
Altyapı	**Dizin**
AIX	*MQ_INSTALLATION_PATH*/java
HP Integrity NonStop Server	*MQ_INSTALLATION_PATH*/opt/mqm/java
HP-UX, Linuxve Solaris	*MQ_INSTALLATION_PATH*/java
Pencereler	*MQ_INSTALLATION_PATH*\java
MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.	

Kuruluş dizini şunları içerir:

- The IBM WebSphere MQ classes for JMS JAR files, which are located in the *MQ_INSTALLATION_PATH*\java\lib directory.
- Java Native Interface 'i kullanan uygulamalar tarafından kullanılan IBM WebSphere MQ yerel kitaplıkları.

  32 bit yerel kitaplıklar, *MQ_INSTALLATION_PATH*\java\lib dizinine kurulur ve 64 bit yerel kitaplıklar *MQ_INSTALLATION_PATH*\java\lib64 dizininde bulunabilir.

  IBM WebSphere MQ yerel kitaplıklarıyla ilgili daha fazla bilgi için bkz. “Java Native Interface (JNI) kitaplıklarının yapılandırılması” sayfa 696.
- “JMS için WebSphere MQ sınıflarıyla sağlanan komut dosyaları” sayfa 762içinde açıklanan ek komut dosyaları. Bu komut dosyaları, *MQ_INSTALLATION_PATH*\java\bin dizininde bulunur.
- JMS API için IBM WebSphere MQ sınıflarına ilişkin belirtimler. Javadoc aracı, API ' nin belirtimlerini içeren HTML sayfalarını oluşturmak için kullanıldı.

  HTML sayfaları *MQ_INSTALLATION_PATH*\java\doc\WMQJMSClasses dizininde yer alıyor.

  UNIX, Linuxve Windows sistemlerinde bu alt dizin, tek tek HTML sayfalarını içerir.
- OGSi desteği. OSGi kod paketleri java \lib\OSGi dizinine kurulur ve “OSGi desteği” sayfa 763içinde anlatılır.
- Herhangi bir JCA 1.5 (ya da sonraki sürümü) uyumlu uygulama sunucusuna yerleştirilebilen IBM WebSphere MQ Resource Adapter.

  IBM WebSphere MQ Resource Adapter, *MQ_INSTALLATION_PATH*\java\lib\jca dizininde bulunur; daha fazla bilgi için bkz. “IBM WebSphere MQ kaynak bağdaştırıcısı” sayfa 704
- Windows 'ta, hata ayıklama için kullanılabilecek simgeler *MQ_INSTALLATION_PATH*\java\lib\simgelerinin dizininde kurulu olmalıdır.

Kuruluş dizininde, diğer IBM WebSphere MQ bileşenlerine ait bazı dosyalar da bulunur. Bu dizinler aşağıdaki gibidir:

- SOAP için JMS iletimi sağlayan SOAP için IBM WebSphere MQ iletimi, *MQ_INSTALLATION_PATH*\java\lib\soap dizinine kurulur. For further information on IBM WebSphere MQ transport for SOAP, see the section of the infocenter describing “SOAP içinWebSphere MQ iletimi” sayfa 908.
- On distributed platforms, the IBM WebSphere MQ Bridge for HTTP is installed in the *MQ_INSTALLATION_PATH*\java\lib\http directory. HTTP için IBM WebSphere MQ köprüsü hakkında daha fazla bilgi için, “HTTP içinWebSphere MQ köprüsü” sayfa 982köprüsünündescribingadlı bölümüne bakın.

JMS için IBM WebSphere MQ sınıflarıyla bazı örnek uygulamalar sağlanır. Çizelge 91 sayfa 693 , her altyapıda örnek uygulamaların kurulu olduğu yeri gösterir.

Çizelge 91. Örnek dizinleri	
Altyapı	**Dizin**
AIX	*MQ_INSTALLATION_PATH*/samp/jms
HP Integrity NonStop Server	*MQ_INSTALLATION_PATH*/opt/mqm/samp/jms
HP-UX, Linuxve Solaris	*MQ_INSTALLATION_PATH*/samp/jms
Pencereler	*MQ_INSTALLATION_PATH*\tools\jms
MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.	

Kuruluştan sonra, uygulamaları derlemek ve çalıştırmak için bazı yapılanış görevlerini gerçekleştirmeniz gerekebilir.

"JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri" sayfa 694 , JMS uygulamaları için basit IBM WebSphere MQ sınıflarını çalıştırmak için gereken sınıf yolunu açıklar. Bu konuda, özel durumlarda gönderme yapılacak ek JAR dosyaları ve JMS için IBM WebSphere MQ sınıflarıyla birlikte verilen komut dosyalarını çalıştırmak üzere ayarlamanız gereken ortam değişkenleri de açıklanmaktadır.

JMS uygulaması için IBM WebSphere MQ sınıflarınızın Java dışındaki dillerde yazılan kodlara bağlanmasını (örneğin, bir Kuyruk Yöneticisine bağlanırken bağ tanımları aktarımında kullanmak için)needsınıfınıza gereksinim duyarsanız, "Java Native Interface (JNI) kitaplıklarının yapılandırılması" sayfa 696 Java komutunun bir değiştirgesi olarak belirtilecek Java Native Interface (JNI) kitaplıklarının yerini nerede bulacağı açıklanır.

Bir uygulamanın izlenmesi ve günlüğe kaydedilmesi gibi özellikleri denetlemek için bir yapılandırma özellikleri dosyası sağlamanız gerekir. JMS yapılandırma özellikleri (properties) dosyasına ilişkin IBM WebSphere MQ sınıfları "IBM WebSphere MQ classes for JMS yapılandırma dosyası" sayfa 698içinde açıklanmıştır.

JMS JAR dosyaları için WebSphere MQ sınıflarının kurulması ve büyütülmesi

The only supported way to get the IBM WebSphere MQ classes for JMS JAR files onto a system, is to install either the IBM WebSphere MQ product, or the WebSphere MQ V7.5 İstemcileri SupportPac-MQC75, or by using a software management tool such as Apache Maven, for more information see "IBM WebSphere MQ classes for JMS ve yazılım yönetimi araçları" sayfa 699.

Bir yazılım yönetimi aracı kullanmıyorsanız, JMS JAR dosyalarına ya da yerel kitaplıklara ya da yerel kitaplıklara ilişkin IBM WebSphere MQ sınıflarını, diğer makinelere ya da JMS için IBM WebSphere MQ sınıflarının kurulu olduğu bir makinede farklı bir konuma taşımayın.

- Düzeltme paketleri, JAR dosyalarının başka bir makineden kopyalandığı bir "kuruluşa" uygulanamaz; bunun nedeni, tüm JAR dosyalarının birbiriyle aynı adımda tutulmasını ve uyumlu düzeylerde olmasını sağlamaktan çok daha zor olur.
- Makineler arasında JMS JAR dosyalarına ilişkin IBM WebSphere MQ sınıflarının kopyalanması, aynı makinede bulunan dosyaların birden çok kopyasına da neden olabilir; bu da kodun bakılmasına ve hata ayıklama sorunlarına neden olabilir.
- The Dspmqver command, used to display version information from an IBM WebSphere MQ installation, only displays version information for the IBM WebSphere MQ classes for JMS installed into the `\java\lib` directory.

  If multiple copies of the files reside on the same machine, running **dspmqver** might not give accurate information about the version of the IBM WebSphere MQ classes for JMS being used by an application.

Uygulama arşivleri içinde (kurumsal uygulama arşivleri ya da EAR dosyaları gibi) JMS JAR dosyalarına ilişkin IBM WebSphere MQ sınıflarını eklemeyin.

- JMS için IBM WebSphere MQ sınıflarında yapılan güncellemeler, bir IBM WebSphere MQ düzeltme paketi kullanılarak uygulanamaz.

- Uygulama tarafından kullanılmakta olan JMS için IBM WebSphere MQ sınıflarının sürümünü kolayca belirlemek IBM Desteği için olanaklı değildir.
- Aynı Java Runtime Environment içinde çalışan birden çok uygulama, JMS için IBM WebSphere MQ sınıflarının farklı sürümlerini içerirse, JMS için IBM WebSphere MQ sınıflarının birden çok sürümü Java Runtime Environment 'a aynı anda yüklenirse, sorunlar ortaya çıkabilir.

  Bu sorunlara örnek olarak aşağıdaki istisnalar yer alır:

```
java.lang.ClassCastException :
com.ibm.mq.jmqi.system.JmqiSystemEnvironment incompatible with
com.ibm.mq.jmqi.system.JmqiSystemEnvironment

java.lang.ClassCastException :
com.ibm.mq.jms.MQQueue incompatible with com.ibm.mq.jms.MQQueue
```

- Bir uygulama, bir kuyruk yöneticisine bağlanmak için BAĞKUR iletisini kullanıyorsa, kuyruk yöneticisine yapılan tüm ana yükseltmeler de, uygulamanın JMS için ilgili IBM WebSphere MQ sınıflarının ilgili düzeyini içerecek şekilde güncellenmesini gerektirir.

  Örneğin, bir kuyruk yöneticisi IBM WebSphere MQ Sürüm 7.5 düzeyine yükseltilirse, BAĞKUR iletisini kullanan kuyruk yöneticisine bağlanan uygulamaların da JMS için IBM WebSphere MQ Sürüm 7.5 sınıflarını içerecek şekilde güncellenmesi gerekir.

JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri

JMS uygulamaları için IBM WebSphere MQ sınıflarını derleyebilmenizi ve çalıştırabilmeniz için, CLASSPATH ortam değişkeninizin ayarının JMS Java arşivi (JAR) dosyasına ilişkin IBM WebSphere MQ sınıflarını içermesi gerekir. Gereksinimlerinize bağlı olarak, sınıf yolunuza başka JAR dosyaları eklemeniz gerekebilir. JMS için IBM WebSphere MQ sınıflarıyla sağlanan komut dosyalarını çalıştırmak için, diğer ortam değişkenleri ayarlanmalıdır.

To compile and run IBM WebSphere MQ classes for JMS applications, use the CLASSPATH setting for your platform as shown in Çizelge 92 sayfa 694. Bu ayar, JMS örnek uygulamaları için IBM WebSphere MQ sınıflarını derleyebilmenizi ve çalıştırabilmeniz için örnekler dizinini içerir. Diğer bir seçenek olarak, ortam değişkenini kullanmak yerine **java** komutundaki sınıf yolunu belirtebilirsiniz.

Çizelge 92. Örnek uygulamalar da içinde olmak üzere, JMS uygulamaları için IBM WebSphere MQ sınıflarını derlemek ve çalıştırmak için CLASSPATH ayarı

Altyapı	**SPATH ayarı**
AIX	CLASSPATH=*MQ_INSTALLATION_PATH*/java/lib/com.ibm.mqjms.jar: *MQ_INSTALLATION_PATH*/samp/jms:
HP Integrity NonStop Server	CLASSPATH=*MQ_INSTALLATION_PATH*/java/lib/com.ibm.mqjms.jar: *MQ_INSTALLATION_PATH*/samp/jms:
HP-UX, Linuxve Solaris	CLASSPATH=*MQ_INSTALLATION_PATH*/java/lib/com.ibm.mqjms.jar: *MQ_INSTALLATION_PATH*/samp/jms:
IBM i	CLASSPATH=/QIBM/ProdData/mqm/java/lib/com.ibm.mqjms.jar: /QIBM/ProdData/mqm/java/samples/jms:
Pencereler	CLASSPATH=*MQ_INSTALLATION_PATH*\java\lib\com.ibm.mqjms.jar; *MQ_INSTALLATION_PATH*\tools\jms;
z/OS	CLASSPATH=*MQ_INSTALLATION_PATH*/mqm/V7R0M0/java/lib/ com.ibm.mqjms.jar: *MQ_INSTALLATION_PATH*/mqm/V6R0M0/java/samples/jms:

Çizelge 92. Örnek uygulamalar da içinde olmak üzere, JMS uygulamaları için IBM WebSphere MQ sınıflarını derlemek ve çalıştırmak için CLASSPATH ayarı (devamı var)	
Altyapı	**SPATH ayarı**
MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.	

com.ibm.mqjms.jar adlı JAR dosyasının bildirgesi, JMS uygulamaları için IBM WebSphere MQ sınıflarının gerektirdiği diğer JAR dosyalarının çoğuna yönelik başvurular içeriyor ve bu JAR dosyalarını sınıf yolunuza eklemeniz gerekmez. Bu JAR dosyaları, bir dizin hizmetinden yönetilen nesneleri ve JTA (Java Transaction API) kullanan uygulamalar tarafından yönetilen nesneleri almak için kullanılan Java Naming and Directory Interface (JNDI) olanağını kullanan uygulamaların gerektirdiği dosyaları içerir.

Ancak, aşağıdaki durumlarda sınıf yolunuza ek JAR dosyaları da eklemelisiniz:

- com.ibm.mq paketinde tanımlanan kanal çıkış arabirimlerini gerçekleştiren kanal çıkış sınıflarını kullanıyorsanız, com.ibm.mq.exits paketinde tanımlananlar yerine, Java JAR dosyasına (com.ibm.mq.jar) ilişkin IBM WebSphere MQ sınıflarını sınıf yolunuza eklemeniz gerekir.
- If you compile your Java code using a Java 2 Software Development Kit (SDK) at Version 1.4.2, you must add the following JAR files to your class path:
  - jms.jar
  - com.ibm.mq.jmqi.jar

  Ayrıca, uygulamanız bir dizin hizmetinden yönetilen nesneleri almak için JNDI kullanıyorsa, aşağıdaki JAR dosyalarını sınıf yolunuza eklemeniz de gerekir:
  - fscontext.jar
  - jndi.jar
  - ldap.jar
  - providerutil.jar

  Uygulamanızın JTA kullanıyorsa, sınıf yolunuza jta.jar öğesini de eklemelisiniz.

  Bu ek JAR dosyalarının, yalnızca uygulamalarınızı derlemek için gerekli olduğunu, bunları çalıştırmak için değil, yalnızca derlemek için gerektiğini unutmayın.

-Xlint seçeneğini kullanarak derleyip derlendiyseniz, com.ibm.mq.ese.jar dosyasının yok olduğunu bildiren bir ileti görebilirsiniz. Uyarıyı yoksayabilirsiniz. Bu dosya yalnızca Extended Security Edition ürününü kurdıysanız bulunur.

JMS için IBM WebSphere MQ sınıflarıyla verilen komut dosyaları aşağıdaki ortam değişkenlerini kullanır:

MQ_JAVA_DATA_PATH
Bu ortam değişkeni, günlük ve izleme çıkışına ilişkin dizini belirtir.

MQ_JAVA_INSTALL_PATH
Bu ortam değişkeni, JMS için WebSphere MQ sınıflarının kurulu olduğu dizini belirtir.

MQ_JAVA_LIB_PATH
Bu ortam değişkeni, JMS kitaplıkları için WebSphere MQ sınıflarının depolandığı dizini belirtir (Çizelge 93 sayfa 696içinde gösterildiği gibi).

Windows' ta, kuruluş sırasında tüm ortam değişkenleri otomatik olarak ayarlanır. Başka bir platform üzerinde, onları kendiniz ayarlamalısınız.

UNIX, HP Integrity NonStop Server, ya da Linux sistemlerinde 32 bit JVM kullanıyorsanız ortam değişkenlerini ayarlamak için, setjmsenv komut kütüğünü kullanabilirsiniz. Bir UNIX ya da Linux sisteminde 64 bit JVM kullanıyorsanız, ortam değişkenlerini ayarlamak için setjmsenv64komut kütüğünü kullanabilirsiniz. Bu komut dosyaları, *MQ_INSTALLATION_PATH*/java/bin dizininde bulunur; burada *MQ_INSTALLATION_PATH* , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.

setjmsenv ya da setjmsenv64 komut dosyasını çeşitli şekillerde kullanabilirsiniz: Çizelgde gösterildiği gibi, gerekli ortam değişkenlerini ayarlamak için temel olarak kullanabilir ya da bir metin düzenleyicisi

kullanarak `.profile` ' a ekleyebilirsiniz. Tipik olmayan bir ayarınız varsa, komut dosyası içeriğini gerektiği gibi düzenleyin. Diğer bir seçenek olarak, komut dosyasını JMS başlangıç komut kütüklerinin çalıştırılacağı her oturumda çalıştırabilirsiniz. If you choose this option you need to run the script in every shell window you start, during the JMS verification process by typing `. ./setjmsenv` or `. ./setjmsenv64`

Java Native Interface (JNI) kitaplıklarının yapılandırılması

JMS uygulamalarına ilişkinIBM WebSphere MQ sınıfları, bağ tanımları iletimi kullanılarak bir kuyruk yöneticisine bağlanan ya da istemci aktarımı kullanılarak bir kuyruk yöneticisine bağlanan ve Java dışındaki dillerde yazılmış kanal çıkış programlarını kullananchannelsınıfları, Java Native Interface (JNI) kitaplıklarına erişen bir ortamda çalıştırılması gerekir.

Bu görev hakkında

Bu ortamı ayarlamak için, Java sanal makinesi (JVM), JMS uygulamasına ilişkin IBM WebSphere MQ sınıflarını başlatmadan önce, Java sanal makinesi (JVM) mqjbnd kitaplığını yükleyebilmesi için ortamın kitaplık yolunu yapılandırmalısınız.

IBM WebSphere MQ , iki Java Native Interface (JNI) kitaplığı sağlar:

mqjbnd
Bu kitaplık, bağ tanımları taşıma özelliğini kullanarak kuyruk yöneticisine bağlanan uygulamalar tarafından kullanılır. Bu arabirim, JMS ve kuyruk yöneticisi için IBM WebSphere MQ sınıfları arasında arabirim sağlar. IBM WebSphere MQ VersionSürüm 7.5 ile kurulan mqjbnd kitaplığı, herhangi bir IBM WebSphere MQ Sürüm 7.5 (ya da önceki) kuyruk yöneticisine bağlanmak için kullanılabilir.

mqjexitstub02
Bir uygulama istemci iletimi kullanılarak bir kuyruk yöneticisine bağlandığında ve Java dışında bir dilde yazılmış bir kanal çıkış programı kullanıyorsa, mqjexitstub02 kitaplığı JMS için IBM WebSphere MQ sınıfları tarafından yüklenir.

Bazı altyapılarda, IBM WebSphere MQ bu JNI kitaplıklarının 32 bit ve 64 bit sürümlerini kurar. Her bir platforma ilişkin kitaplıkların yeri Tablo 1içinde gösterilir.

Çizelge 93. Her platform için JMS kitaplıklarına ilişkin IBM WebSphere MQ sınıflarının yeri

Altyapı	**JMS kitaplıkları için IBM WebSphere MQ sınıflarını içeren dizin**
AIX	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)
HP-UX	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)
Linux (güç , x86-64 ve zSeries s390x platformları)	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)
Linux (x86 platformu) Linux (zSeries altyapısı)	*MQ_INSTALLATION_PATH* /java/lib
Solaris (x86-64 ve SPARC platformları)	*MQ_INSTALLATION_PATH*/java/lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*/java/lib64 (64 bit kitaplık)

Çizelge 93. Her platform için JMS kitaplıklarına ilişkin IBM WebSphere MQ sınıflarının yeri (devamı var)	
Altyapı	**JMS kitaplıkları için IBM WebSphere MQ sınıflarını içeren dizin**
Pencereler	*MQ_INSTALLATION_PATH*\java\lib (32 bit kitaplık) *MQ_INSTALLATION_PATH*\java\lib64 (64 bit kitaplık)
MQ_INSTALLATION_PATH , IBM WebSphere MQ ' in kurulu olduğu üst düzey dizini temsil eder.	

Yordam

1. JVM ' nin **java.library.path** özelliğini yapılandırın; bu, şu iki şekilde yapılabilir:
   - Aşağıdaki örnekte gösterildiği gibi, JVM bağımsız değişkenini belirterek:

     ```
     -Djava.library.path=<path_to_library_directory>
     ```

     Linux Örneğin, varsayılan konum kuruluşu için Linux üzerindeki bir 64 bit JVM için şunu belirtin:

     ```
     -Djava.library.path=/opt/mqm/java/lib64
     ```

   - Shell 'in ortamını yapılandırarak, JVM kendi `java.library.path`' ı ayarlayacak. Bu yol, platforma göre ve IBM WebSphere MQürününü kurduğunuz yer tarafından değişir. Örneğin, 64 bit JVM ve varsayılan bir IBM WebSphere MQ kuruluş yeri için aşağıdaki ayarları kullanabilirsiniz:

     AIX
     ```
     export LIBPATH=/usr/mqm/java/lib64:$LIBPATH
     ```

     Solaris HP-UX Linux
     ```
     export LD_LIBRARY_PATH=/opt/mqm/java/
     lib64:$LD_LIBRARY_PATH
     ```

     Windows
     ```
     set PATH=C:\Program Files\IBM\MQ\java\lib64;%PATH%
     ```

   Ortam doğru şekilde yapılandırılmadığında gördüğünüz kural dışı durum yığınına bir örnek aşağıdaki gibidir:

```
Nedeni: com.ibm.mq.jmqi.local.LocalMQ$4: CC=2;RC=2495;
AMQ8598:  WebSphere MQ yerel JNI kitaplığının yüklenmesi başarısız oldu: 'mqjbnd'.
    com.ibm.mq.jmqi.local.LocalMQ.loadLib(LocalMQ.java:1268)
    com.ibm.mq.jmqi.local.LocalMQ$1.run(LocalMQ.java:309)
    java.security.AccessController.doPrivileged(AccessController.java:400)
    com.ibm.mq.jmqi.local.LocalMQ.initialise_inner(LocalMQ.java:259)
    com.ibm.mq.jmqi.local.LocalMQ.initialise: (LocalMQ.java:221)
    com.ibm.mq.jmqi.local.LocalMQ. < init> (LocalMQ.java:1350)
    at com.ibm.mq.jmqi.local.LocalServer.<init>(LocalServer.java:230)
    sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Yönteminde)

sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:86)


sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.jav
a:58)
    java.lang.reflect.Constructor.newInstance(Constructor.java:542)
    com.ibm.mq.jmqi.JmqiEnvironment.getInstance: (JmqiEnvironment.java:706)
    com.ibm.mq.jmqi.JmqiEnvironment.getMQI(JmqiEnvironment.java:640)
    at
com.ibm.msg.client.wmq.factories.WMQConnectionFactory.createV7ProviderConnection(WMQConnectionF
actory.java:8437)
    ... 7 daha fazla
Nedeni: java.lang.UnsatisfiedLinkError: mqjbnd ( java.library.pathiçinde bulunamadı)
    java.lang.ClassLoader.loadLibraryWithPath(ClassLoader.java:1235)
    java.lang.ClassLoader.loadLibraryWithClassLoader(ClassLoader.java:1205)
    java.lang.System.loadLibrary(System.java:534)
    com.ibm.mq.jmqi.local.LocalMQ.loadLib(LocalMQ.java:1240)
    ... 20 tane daha
```

2. 32 bit ya da 64 bit ortam ayarlandıktan sonra, aşağıdaki komutu kullanarak JMS uygulamasına ilişkin IBM WebSphere MQ sınıflarını başlatın:

   ```
   java application-name
   ```

   Burada *uygulama-adı* , JMS uygulamasının çalıştırılacağı IBM WebSphere MQ sınıflarının adıdır.

   Aşağıdaki durumlarda JMS için IBM WebSphere MQ sınıfları tarafından IBM WebSphere MQ Neden Kodu 2495 (MQRC_MODULE_NOT_FOUND) içeren bir kural dışı durum oluşur:

   - 32 bit Java Runtime Environment 64 bit Java Yerel Kitaplığı 'nı yükleyemediğinden, JMS uygulaması için IBM WebSphere MQ sınıfları 32 bit Java çalıştırma zamanı ortamında çalıştırılır ve JMS için IBM WebSphere MQ sınıfları için 64 bit bir ortam ayarlanmıştır.
   - 64 bit Java Runtime Environment 32 bit Java Yerel Kitaplığı 'nı yükleyemediğinden, JMS uygulaması için IBM WebSphere MQ sınıfları 64 bit Java çalıştırma zamanı ortamında çalıştırılır ve JMS için IBM WebSphere MQ sınıfları için 32 bit bir ortam ayarlanmıştır.

IBM WebSphere MQ classes for JMS yapılandırma dosyası

JMS yapılandırma dosyası için bir WebSphere MQ sınıfları, JMS için WebSphere MQ sınıflarını yapılandırmak için kullanılan özellikleri belirtir.

JMS yapılandırma dosyası için bir WebSphere MQ sınıflarının biçimi, standart bir Java özellikler (properties) dosyası biçimidir. JMS kuruluş dizinine ilişkin WebSphere MQ sınıflarının `bin` alt dizininde jms.config adlı örnek bir yapılanış kütüğü sağlanır. Bu dosya, desteklenen tüm özellikleri ve bunların varsayılan değerlerini belgeler.

JMS yapılandırma dosyası için bir WebSphere MQ sınıflarının adını ve konumunu seçebilirsiniz. Uygulamanıza başladığınızda, aşağıdaki biçimi kullanarak bir **java** komutunu kullanın:

```
java -Dcom.ibm.msg.client.config.location=config_file_url application_name
```

config_file_url komutunda, JMS yapılandırma dosyasına ilişkin WebSphere MQ sınıflarının adını ve yerini belirten bir URL (uniform resource locator; URL). Şu tiplerin URL ' leri desteklenir: http, file, ftp ve jar.

Aşağıda bir **java** komutu örneği yer alıyor:

```
java -Dcom.ibm.msg.client.config.location=file:/D:/mydir/myjms.config MyAppClass
```

Bu komut, yerel Windows sisteminde D:\mydir\mjms.config dosyası olarak JMS yapılandırma dosyası için WebSphere MQ sınıflarını tanıtır.

Bir uygulama başlatıldığında, JMS için WebSphere MQ sınıfları, yapılandırma dosyasının içeriğini okur ve belirtilen özellikleri bir iç özellik deposunda saklar. **java** komutu bir yapılandırma dosyasını belirtmezse ya da yapılandırma dosyası bulunamazsa, JMS için WebSphere MQ sınıfları tüm özellikler için varsayılan değerleri kullanır. If required, you can override any property in the configuration file by specifying it as a system property on the **java** command.

Bir uygulama ile kuyruk yöneticisi ya da aracı arasındaki desteklenen aktarımlardan herhangi biriyle JMS yapılandırma dosyası için bir WebSphere MQ sınıfları kullanılabilir.

JMS yapılandırma dosyasına ilişkin WebSphere MQ sınıflarında bir özellik ayarlayarak başlatma izlemeyi belirtemezsiniz. Başlatma izlemeyi yalnızca, aşağıdaki örnekte gösterildiği gibi, **java** komutunda bir sistem özelliği ayarlayarak belirtebilirsiniz:

```
java -Dcom.ibm.msg.client.commonservices.trace.startup=true
     -Dcom.ibm.msg.client.config.location=file:/D:/mydir/myjms.config
      MyAppClass
```

Bir WebSphere MQ MQI istemcisi yapılanış kütüğünde belirtilen özelliklerin geçersiz kılınması

Bir WebSphere MQ MQI istemcisi yapılanış dosyası, JMS için WebSphere MQ sınıflarını yapılandırmak için kullanılan özellikleri de belirtebilir. Ancak, bir WebSphere MQ MQI istemcisi yapılanış dosyasında belirtilen özellikler yalnızca, bir uygulama istemci kipinde bir kuyruk yöneticisine bağlandığında geçerlidir.

Gerekirse, JMS yapılandırma dosyasına ilişkin bir WebSphere MQ sınıflarında bir özellik olarak belirterek, WebSphere MQ MQI istemcisi yapılanış dosyasındaki herhangi bir özniteliği geçersiz kılabilirsiniz. To override an attribute in a WebSphere MQ MQI client configuration file, use an entry with the following format in the WebSphere MQ classes for JMS configuration file:

```
com.ibm.mq.cfg.stanza.propName=propValue
```

Girdideki değişkenler aşağıdaki anlamlara sahiptir:

stanza
The name of the stanza in the WebSphere MQ MQI client configuration file that contains the attribute

propName
The name of the attribute as specified in the WebSphere MQ MQI client configuration file

propValue
WebSphere MQ MQI istemcisi yapılanış kütüğünde belirtilen özniteliğin değerini geçersiz kılan özelliğin değeri

Diğer bir seçenek olarak, özelliği **java** komutunda sistem özelliği olarak belirterek bir WebSphere MQ MQI istemcisi yapılanış dosyasındaki bir özniteliği geçersiz kılabilirsiniz. Özelliği bir sistem özelliği olarak belirtmek için önceki biçimi kullanın.

Bir WebSphere MQ MQI istemcisi yapılandırma dosyasında yalnızca aşağıdaki öznitelikler, JMS için WebSphere MQ sınıflarıyla ilgilidir. Diğer öznitelikleri belirtmiş ya da geçersiz kıyorsanız, bu herhangi bir etkisi olmaz.

Stanza	Öznitelik
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultYolu
ClientExitİstemci yapılandırma dosyasının yol gösterişi	ExitsDefaultPath64
ClientExitİstemci yapılandırma dosyasının yol gösterişi	JavaExitsClasspath
İstemci yapılandırma dosyasınınMessageBuffer kısmı	MaximumSize
İstemci yapılandırma dosyasınınMessageBuffer kısmı	PurgeTime
İstemci yapılandırma dosyasınınMessageBuffer kısmı	UpdatePercentage
İstemci yapılandırma dosyasının TCP stanzası	ClntRcvBufSize
İstemci yapılandırma dosyasının TCP stanzası	ClntSndBufSize
İstemci yapılandırma dosyasının TCP stanzası	Bağlantı_Zamanaşımı
İstemci yapılandırma dosyasının TCP stanzası	KeepAlive

IBM WebSphere MQ classes for JMS ve yazılım yönetimi araçları

Apache Maven gibi yazılım yönetimi araçları IBM WebSphere MQ classes for JMSile birlikte kullanılabilir.

Birçok büyük geliştirme kuruluşu, bu araçları, üçüncü kişi kitaplıklarının havuzlarını merkezi olarak yönetmek için kullanır.

IBM WebSphere MQ classes for JMS , JAR dosyalarından oluşan bir sayıdan oluşur. Bu API ' yi kullanarak Java dili uygulamaları geliştirirken, uygulamanın geliştirilmekte olduğu makinede bir IBM WebSphere MQ Server, Client ya da Client SupportPac kuruluşu gereklidir.

Bu tür bir aracı kullanmak ve IBM WebSphere MQ classes for JMS ' yi merkezi olarak yönetilen bir havuza oluşturan JAR dosyalarını eklemek istiyorsanız, aşağıdaki noktalar gözlenmelidir:

- Bir havuz ya da taşıyıcı, yalnızca kuruluşunuz içindeki geliştiriciler tarafından kullanılabilir hale getirilmelidir. Kuruluşun dışındaki herhangi bir dağılıma izin verilmez.
- Havuzun tek bir IBM WebSphere MQ yayınından ya da Düzeltme Paketinden eksiksiz ve tutarlı bir JAR dosyası kümesi içermesi gerekir.
- Havuzu, IBM Desteği tarafından sağlanan herhangi bir bakım ile güncellemekle göreviniz vardır.

IBM WebSphere MQ Version 7.5için, havuza aşağıdaki JAR dosyalarının kurulması gerekir:

- com.ibm.mqjms.jar.
- com.ibm.mq.jar
- com.ibm.mq.jmqi.jar
- com.ibm.mq.pcf.jar
- com.ibm.mq.headers.jar
- IBM WebSphere MQ classes for JMSkullanıyorsanız,CL3Export.jar gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,CL3Nonexport.jar gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,jndi.jar gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,ldap.jar dosyası gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,rmm.jar gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,dhbcore.jar gereklidir.
- IBM WebSphere MQ classes for JMSkullanıyorsanız,jms.jar gereklidir.
- IBM WebSphere MQ classes for JMS kullanıyorsanız ve bir dosya sistemi JNDI bağlamında saklanan JMS denetimli nesnelere erişiyorsanız,fscontext.jar gereklidir.
- IBM WebSphere MQ classes for JMS kullanıyorsanız ve bir dosya sistemi JNDI bağlamında saklanan JMS denetimli nesnelere erişiyorsanız,providerutil.jar .

Java güvenlik yöneticisi altındaki JMS uygulamaları için WebSphere MQ sınıflarının çalıştırılması

WebSphere MQ classes for JMS can run with the Java security manager enabled. Güvenlik yöneticisini etkinleştiren uygulamaları başarıyla çalıştırmak için, Java sanal makinenizi (JVM) uygun bir ilke yapılandırma dosyasıyla yapılandırmalısınız.

Bunun en basit yolu, JRE ' nizle birlikte sağlanan ilke yapılandırma dosyasını değiştirmemeniz. Çoğu sistemde, bu dosya JRE dizininize göre `lib/security/java.policy`yolunda saklanır. İlke yapılandırma dosyasını tercih ettiğiniz düzenleyiciyi ya da JRE ' nizle birlikte sağlanan `policytool` programını kullanarak düzenleyebilirsiniz.

Önemli: V 7.5.0.8 Mümkün olan her yerde, *allowlist* terimi, *beyaz liste*terimini değiştirdi. Bir kural dışı durum, aşağıdaki Java sistemi özellik adlarından biridir.

Uygulamanızı içeren Java güvenlik yöneticisi düzeneğini kullanırsanız, aşağıdaki izinleri vermeniz gerekir:

- FilePermission on any allowlist file that you use, with read permission for ENFORCEMENT mode, write permission for DISCOVER mode.
- com.ibm.mq.jms.whitelist, com.ibm.mq.jms.whitelist.discoverve com.ibm.mq.jms.whitelist.mode özelliklerindePropertyPermission (okuma).

ClassName allowlistesi, APAR IT14385 ve IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 8ile desteklenmektedir. Daha fazla bilgi için, bkz. “JMS ObjectMessage' daClassName allowlisteleme” sayfa 701.

Aşağıda, JMS için WebSphere MQ sınıflarının varsayılan güvenlik yöneticisi altında başarıyla çalışabilmesi için izin veren bir ilke yapılandırma dosyasında iki giriş örneği yer alıyor:

```
grant codeBase "file:MQ_INSTALLATION_PATH/java/lib/com.ibm.mq.jmqi.jar" {
  //Required
  permission java.util.PropertyPermission "user.name","read";
  permission java.util.PropertyPermission "os.name","read";
  //Required if mqclient.ini/mqs.ini configuration files are used
  permission java.io.FilePermission "/var/mqm/mqclient.ini","read";
  permission java.io.FilePermission "/var/mqm/mqs.ini","read";
  //For the client transport type.
  permission java.net.SocketPermission "*","connect";
  //For the bindings transport type.
  permission java.lang.RuntimePermission "loadLibrary.*";
  //For applications that use CCDT tables (access to the CCDT
AMQCLCHL.TAB)
  permission java.io.FilePermission
"/var/mqm/qmgrs/QMGR/@ipcc/AMQCLCHL.TAB","read";
  //For applications that use User Exits
  permission java.io.FilePermission "/var/mqm/exits/*","read";
  permission java.lang.RuntimePermission "createClassLoader";
  //Required for the z/OS platform
  permission java.util.PropertyPermission
"com.ibm.vm.bitmode","read";
};
grant codeBase "file:MQ_INSTALLATION_PATH/java/lib/com.ibm.mqjms.jar" {
  permission java.util.PropertyPermission "user.name","read";
  permission java.util.PropertyPermission "os.name","read";
  permission java.util.PropertyPermission "console.encoding","read";
  permission java.lang.RuntimePermission "setContextClassLoader";
  //tracing permissions
  permission java.util.PropertyPermission "com.ibm.msg.client.commonservices.*","read";
  permission java.util.PropertyPermission "MQJMS_TRACE_LEVEL","read";
  permission java.util.logging.LoggingPermission "control";
  //Wherever trace output is expected
  permission java.io.FilePermission "/tmp/*","read,write";
  //Required for the z/OS platform
  permission java.util.PropertyPermission
"com.ibm.vm.bitmode","read";
};
```

Notlar:

- *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.
- İlk `grant` deyimi, JMS için WebSphere MQ sınıflarının gerektirdiği izinleri ve ikinci `grant` deyimi, JMS uygulaması için bir WebSphere MQ sınıfları tarafından gereken izinleri içerir.
- JMS için WebSphere MQ sınıflarının bir uygulamanın Java arşiv (JAR) dosyalarına erişmesine izin vermek için, ilk `grant` deyime aşağıdaki izni ekleyin:

  ```
  permission java.io.FilePermission "/path_to_your_app/-", "read";
  ```

- Bu `grant` deyimlerini ilke yapılandırma dosyanızın içinde kullanmak için, yol adlarını JMS için WebSphere MQ sınıflarını kurmuş olduğunuz yere ve uygulamalarınızı sakladığınız yere bağlı olarak değiştirmeniz gerekebilir.
- JMS için WebSphere MQ sınıflarıyla sağlanan örnek uygulamalar ve bunları çalıştırmak için komut dosyaları, güvenlik yöneticisini etkinleştirmez.

V 7.5.0.8 JMS ObjectMessage' daClassName allowlisteleme

V 7.5.0.8 JMS için WebSphere MQ sınıflarında, JMS ObjectMessage arabiriminin uygulanmasında sınıfların ödenek listesi desteği, Java nesnesi serileştirme ve dizisel biçimden geri çevirme düzeneği ile ilgili olabilecek bazı güvenlik risklerine karşı bir azaltma sağlar.

Not: Mümkün olan her yerde, *allowlist* terimi, *beyaz liste*terimini değiştirdi. Bir kural dışı durum, bu konuda belirtilen Java sistem özelliği adlarından biridir.

The Java object serialization and deserialization mechanism has been identified as a potential security risk because deserialization instantiates arbitrary Java objects, where there is the potential for maliciously sent data to cause various problems. One notable application of serialization is in Java Message Service (JMS) ObjectMessages that use serialization to encapsulate and transfer arbitrary objects.

Serileştirme allowing, serileştirme pozları oluşturan bazı risklere karşı, olası bir hafiflettirme biçimidir. By explicitly specifying which classes can be encapsulated in, and extracted from, ObjectMessages, allowlisting provides some protection against some serialization risks.

JMS için WebSphere MQ sınıflarına izin listesi

APAR IT14385 ve IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 8ile, JMS için WebSphere MQ sınıfları, JMS ObjectMessage arabiriminin uygulanmasında sınıfların allowcasting özelliğini destekler. Allowlist, ObjectMessage.setObject() ile hangi Java sınıflarının diziselleştirilebileceğini ve ObjectMessage.getObject() ile dizisel biçimden geri çevrilebilir olduğunu tanımlar.

Attempts to serialize or deserialize an instance of a class not included in the allowlist with ObjectMessage cause a javax.jms.MessageFormatException to be thrown, with a java.io.InvalidClassException as its cause.

Allowlist oluşturma

Önemli: JMS için WebSphere MQ sınıfları, allowlist ile dağıtılamaz. ObjectMessages kullanılarak aktarılacak sınıfların seçimi bir uygulama tasarım seçeneği ve IBM WebSphere MQ bunu önceden muaf olarak gösteremez.

Bu nedenle, allowlocting mekanizması, iki işlem kipinin yapılmasına olanak sağlar:

Keşif
In this mode, the mechanism produces a listing of fully qualified class names, reporting all classes that have been observed to be serialized or deserialized in ObjectMessages.

Zorlama
Bu kipte, mekanizma, allowlist 'te olmayan sınıfları diziselleştirme ya da dizisel biçimden geri çevirme girişimlerini reddederek, allowlist (allowing) listesini uygular.

Bu mekanizmayı kullanmak istiyorsanız, başlangıçta diziselleştirilmiş ve dizisel biçimden geri çevrilen sınıfların listesini toplamak, listeyi gözden geçirmek ve allowlisteniz için bunu temel olarak kullanmak için DISPLAY (DISCOVERY) kipinde çalışmalısınız. Listenin değiştirilmeden kullanılması uygun olabilir, ancak bu işlemi yapmaya karar vermeden önce listenin ilk önce gözden geçirilmesi gerekir.

Allowlisteleme mekanizmasını denetleme

Allowlisteleme mekanizmasını denetlemek için kullanılabilecek üç sistem özelliği vardır:

com.ibm.mq.jms.whitelist
Bu özellik aşağıdaki yollardan biriyle belirtilebilir:

- Allowlist (allowlist) içeren dosyanın yol adı, dosya URI biçiminde (`file:` ile başlayarak). DISABLE kipinde bu dosya, allowlicting mekanizmasının içine yazılır. Dosya var olmamalıdır. Dosya varsa, mekanizma bu dosyanın üzerine yazmak yerine bir kural dışı durum yayınlar. UYGULAMA kipinde bu dosya, allowlisteleme mekanizması tarafından okunurdur.
- Allowlist (allowlist) oluşturan tam olarak nitelenmiş sınıf adlarından oluşan virgülle ayrılmış.

Bu özellik ayarlanmamış ise, allowlist mekanizması etkin değildir.

Bir Java güvenlik yöneticisi kullanıyorsanız, JMS JAR dosyaları için WebSphere MQ sınıflarının bu dosyaya okuma ve yazma erişimine sahip olduğundan emin olmanız gerekir.

com.ibm.mq.jms.whitelist.discover

- Bu özellik ayarlanmazsa ya da false değerine ayarlanırsa, allowlist düzeneği UYGULAMA kipinde çalışır.

- Bu özellik true (doğru) olarak ayarlanırsa ve allowlist bir dosya URI 'si olarak belirtilmişse, allowlist mekanizması DISABLE kipi içinde çalışır.
- Bu özellik doğru olarak ayarlanırsa ve allowlist, sınıf adları listesi olarak belirtilmişse, allowlist mekanizması uygun bir kural dışı durum oluşturur.
- Bu özellik true olarak ayarlanırsa ve allowlist com.ibm.mq.jms.whitelist özelliğikullanılarak belirtilmediyse, allowlist mekanizması etkin değildir.
- Bu özellik true (doğru) olarak ayarlanırsa ve allowlist dosyası önceden varsa, allowlist düzeneği bir java.io.InvalidClassException yayınlar ve bu girişlere dosya eklenmez.

com.ibm.mq.jms.whitelist.mode
Bu dizgi özelliği şu üç yöntemden herhangi birinde belirtilebilir:

- Bu özellik SERIALIZE olarak ayarlandıysa, UYGULAMA kipi yalnızca ObjectMessage.setObject() yönteminde allowlist geçerlilik denetimini gerçekleştirir.
- Bu özellik DESERALIZE olarak ayarlandıysa, UYGULAMA kipi yalnızca ObjectMessage.getObject() yönteminde allowlist geçerlilik denetimini gerçekleştirir.
- Bu özellik ayarlanmazsa ya da başka bir değere ayarlanmışsa, UYGULAMA kipi hem ObjectMessage.getObject() hem de ObjectMessage.setObject() yöntemlerinde allowlist geçerlilik denetimi gerçekleştirir.

Allowlist dosyasının biçimi

Bunlar, allowlist dosyasının biçiminin ana özellikleridir:

- Allowlist dosyası, platforma uygun satır sonlarıyla birlikte varsayılan platform dosyası kodlamasında yer alıyor.

  Not: Türdeş olmayan sistemler arasında geçiş yapmak için dosya dönüştürmeye gerek duyabilir.
- Boş olmayan her bir satır, tam olarak nitelenmiş bir sınıf adı içerir. Boş satırlar yoksayılır.
- Açıklamalar, '#' karakterini izleyen herhangi bir şey, satırın sonuna kadar eklenebilir ve yoksayılır.
- Çok temel bir karalama mekanizması vardır:
  - '*', bir sınıf adının **son** öğesi olabilir.
  - '*', bir sınıf adının **tek** öğesiyle eşleşir; yani, sınıfın bir parçası değil, ancak bu öğedir.

  So `com.ibm.mq.*` would match `com.ibm.mq.MQMessage` but not `com.ibm.mq.jmqi.remote.api.RemoteFAP`.

  Genel, belirtik bir paket adı olmayan sınıflara ilişkin varsayılan paketteki sınıflar için çalışmazsa, "*" sınıf adı reddedilir.
- Hatalı biçimlendirilmiş allowlist dosyaları; örneğin, `com.ibm.mq.*.Message`gibi bir girdiyi içeren, genel arama karakterinin son öğe olmadığı dosyalar, bir java.lang.IllegalArgumentException ' nin atılmasına neden olur.
- Boş bir allowlist dosyası, ObjectMessagekullanımını tamamen devre dışı bırakmanın etkisine sahiptir.

Allowlist 'in biçimi virgülle ayrılmış bir liste olarak biçimlendirir

Aynı genel arama mekanizması, virgülle ayrılmış bir liste olarak bir allowlist için kullanılabilir.

- '*', bir komut satırında ya da bir kabuk komut dosyasında ya da toplu iş dosyasında belirtildiyse, işletim sistemi tarafından genişletilebilir, bu nedenle özel işleme gerekebilir.
- '#' açıklama karakteri yalnızca bir dosya belirtildiğinde geçerlidir. Allowlist, sınıf adlarının virgülle ayrılmış bir listesi olarak belirtilirse, işletim sisteminin ya da kabuğun bunu işlemediğini varsayarsak, birçok UNIX ya da Linux kabuğunda varsayılan yorum karakteri olduğu için, normal bir karakter olarak işlem görür.

Ödenek listesi ne zaman olur?

Uygulama ilk olarak bir ObjectMessage setMessage() ya da getMessage() yöntemini çalıştırdığında ödenek listesi başlatılır.

Sistem özellikleri değerlendirilir, allowlist dosyası açılır ve EXTACY (uygulama) kipinde, mekanizmanın kullanıma hazırlandığında, allowlistelenen sınıfların listesi yüklenir. Bu noktada, uygulama için IBM WebSphere MQ JMS günlük dosyasına bir giriş yazılır.

Mekanizma ilk kullanıma hazırlandığında, parametreleri değişmeyebilir. Kullanıma hazırlama süresi, uygulama davranışına bağlı olduğu için kolay tahmin edilmemektedir. Bu nedenle, sistem özelliği ayarları ve allowlist dosya içeriği, uygulamanın başlatıldığı zamandan itibaren değişmez olarak kabul edilmelidir. Sonuçlar garanti edilmediği için, uygulama çalışırken allowlist dosyasının özelliklerini ya da içeriğini değiştirmeyin.

Dikkate alınacak noktalar

The best approach to mitigating the risks intrinsic to the Java serialization mechanism would be to explore alternative approaches to data transfer such as using JSON instead of ObjectMessage. Using IBM WebSphere MQ Advanced Message Security (AMS) mechanisms can add further security by ensuring that messages come from trusted sources.

Uygulamanızı içeren Java güvenlik yöneticisi düzeneğini kullanırsanız, aşağıdaki izinleri vermeniz gerekir:

- FilePermission on any allowlist file that you use, with read permission for ENFORCEMENT mode, write permission for DISCOVER mode.
- com.ibm.mq.jms.whitelist, com.ibm.mq.jms.whitelist.discoverve com.ibm.mq.jms.whitelist.mode özelliklerindePropertyPermission (okuma).

İlgili kavramlar

“Java güvenlik yöneticisi altındaki JMS uygulamaları için WebSphere MQ sınıflarının çalıştırılması” sayfa 700

WebSphere MQ classes for JMS can run with the Java security manager enabled. Güvenlik yöneticisini etkinleştiren uygulamaları başarıyla çalıştırmak için, Java sanal makinenizi (JVM) uygun bir ilke yapılandırma dosyasıyla yapılandırmalısınız.

IBM WebSphere MQ kaynak bağdaştırıcısı

Kaynak bağdaştırıcısı, uygulama sunucusunda çalışan uygulamaların IBM WebSphere MQ kaynaklarına erişmelerini sağlar. Gelen ve giden iletişimi destekler.

Java Platform, Enterprise Edition (Java EE) Connector Architecture (JCA), Java EE ortamında çalışan uygulamaların IBM WebSphere MQ ya da Db2gibi bir Kurumsal Bilgi Sistemine (EIS) çalıştırılması için standart bir yol sağlar. IBM WebSphere MQ kaynak bağdaştırıcısı JCA 1.5 arabirimlerini gerçekleştirir ve IBM WebSphere MQ classes for JMSöğesini içerir. Bir uygulama sunucusunda çalışan JMS uygulamalarına ve ileti odaklı Bean 'lere (MDBs), bir IBM WebSphere MQ kuyruk yöneticisinin kaynaklarına erişmenizi sağlar. Kaynak bağdaştırıcısı, hem noktadan noktaya iletişim etki alanını, hem de yayınlama/abone olma etki alanını destekler.

IBM WebSphere MQ kaynak bağdaştırıcısı, bir uygulama ile kuyruk yöneticisi arasında iki iletişim tipini destekler:

Giden iletişim
Bir uygulama kuyruk yöneticiyle bağlantı başlatır ve JMS iletilerini JMS hedeflerine gönderir ve JMS hedeflerini zamanuyumlu bir şekilde alır.

Gelen iletişim
Bir JMS hedefine gelen bir JMS iletisi, iletiyi zamanuyumsuz olarak işleyen bir MDB ' ye teslim edilir.

IBM WebSphere MQ classes for JMSile ilgili daha fazla bilgi için bkz. “JMS için WebSphere MQ sınıflarının kullanılması” sayfa 687.

Kaynak bağdaştırıcısı IBM WebSphere MQ classes for Javadeğerini de içerir. Sınıflar otomatik olarak, kaynak bağdaştırıcısının konuşlandırıldığı bir uygulama sunucusunda çalışan uygulamalara otomatik olarak sunulur ve bu uygulama sunucusunda çalışan uygulamaların bir IBM WebSphere MQ kuyruk yöneticisinin kaynaklarına erişirken IBM WebSphere MQ classes for Java API ' yı kullanmasına olanak sağlar. IBM WebSphere MQ classes for Javaile ilgili daha fazla bilgi için bkz. “Java için WebSphere MQ sınıflarının kullanılması” sayfa 627.

The use of the IBM WebSphere MQ classes for Java within a Java EE environment is supported with restrictions. Bu kısıtlamalar hakkında bilgi için bkz. “Java platformu Enterprise Editioniçindeki Java uygulamaları için IBM WebSphere MQ sınıfları çalıştırılıyor” sayfa 685.

JCA kaynak bağdaştırıcısını desteklemek için gereken diğer belgeler

JCA kaynak bağdaştırıcısının nasıl yapılandırılabileceği hakkında bilgi için uygulama sunucunuzun belgelerine bakın.

Her uygulama sunucusu, kendi denetim arabirimleri kümesini sağlar. Bazı uygulama sunucuları, JCA kaynaklarını tanımlamak için grafik kullanıcı arabirimleri sağlar, ancak diğer kullanıcılar denetimcinin XML konuşlandırma planlarını yazmasını gerektirir. Bu nedenle, WebSphere MQ kaynak bağdaştırıcısının her bir uygulama sunucusu için nasıl yapılandırılacağı hakkında bilgi sağlamak için bu belge kapsamı dışındadır. Bu belge yalnızca konfigürasyonunu tanımlamanız gerekenlerle odaklanır. JCA kaynak bağdaştırıcısının nasıl yapılandırılabilmesiyle ilgili bilgi için uygulama sunucunuzla birlikte sağlanan belgelere bakın.

Bu belgeleri anlamak için JMS ' ye ilişkin JMS ve WebSphere MQ sınıflarıyla ilgili bilgi sahibi olmanız gerekir. WebSphere MQ kaynak bağdaştırıcısının konfigürasyonunu tanımlamak için kullanılan özelliklerin çoğu, JMS nesneleri için WebSphere MQ sınıflarının özelliklerine eşdeğerdir ve aynı işleve sahiptir.

WebSphere MQ Resource Adapter ürününün kuruluşu

WebSphere MQ kaynak bağdaştırıcısı bir kaynak arşivi (RAR) dosyası olarak sağlanır. Uygulama sunucunuzda RAR dosyasını kurun. Dizin eklemek için sistem yoluna dizin eklemeniz gerekebilir.

WebSphere MQ kaynak bağdaştırıcısı, wmq.jmsra.raradlı bir kaynak arşivi (RAR) dosyası olarak sağlanır. This file is installed with WebSphere MQ classes for JMS in the directory shown in Çizelge 94 sayfa 705.

Çizelge 94. Her platform için wmq.jmsra.rar dosyası içeren dizin

Altyapı	**Dizin**
AIX, HP-UX, Linux ve Solaris	*MQ_INSTALLATION_PATH* /java/lib/jca
IBM i	/QIBM/ProdData/mqm/java/lib/jca
Pencereler	*MQ_INSTALLATION_PATH* \java\lib\jca
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.	

RAR dosyası, JMS için WebSphere MQ sınıfları ve JCA arabirimlerinin WebSphere MQ somutlaması içerir.

You must install the WebSphere MQ resource adapter RAR file in your application server, but the way you do this depends on the application server. Kaynak bağdaştırıcısı RAR dosyasının nasıl kurulacağı hakkında bilgi için uygulama sunucunuza ilişkin belgelere bakın.

UNIX and Linux sistemlerindeki bağ tanımları bağlantıları için, Java Native Interface (JNI) kitaplıklarını içeren dizinin sistem yolunda olduğundan emin olmanız gerekir. Bu dizinin, JMS kitaplıklarına ilişkin WebSphere MQ sınıflarını da içeren konumu için bkz. Çizelge 93 sayfa 696. Windowsüzerinde, JMS için WebSphere MQ sınıflarının kuruluşu sırasında bu dizin otomatik olarak sistem yoluna eklenir.

İşlemler hem istemci, hem de bağ tanımları kipinde desteklenir.

Kaynak bağdaştırıcısı tarafından kullanılan JMS için WebSphere MQ kaynak bağdaştırıcısı ve WebSphere MQ sınıflarının sürümü aynı yayın düzeyinde olmalıdır.

WebSphere Application Server ve WebSphere MQ kaynak bağdaştırıcısı
WebSphere MQ kaynak bağdaştırıcısını WebSphere Application Server Sürüm 6 ile kullanmayın. WebSphere Application Server, V7, WebSphere MQ V7 Resource Adapter ürününün bir sürümünü içerir.

WebSphere Application Server, V6içinde WebSphere MQ kaynak bağdaştırıcısını kullanmayın. Bir JMS uygulamasından WebSphere Application Server içinden bir WebSphere MQ kuyruk yöneticisinin kaynaklarına erişmek için, WebSphere MQ ileti alışverişi sağlayıcısını kullanın. WebSphere MQ ileti alışverişi sağlayıcısı, JMS için WebSphere MQ sınıflarının bir sürümünü içerir.

WebSphere Application Server, V7, WebSphere MQ V7 Resource Adapter ürününün bir sürümünü içerir.

Daha fazla bilgi için bkz. teknik not WebSphere MQ Resource Adapter (RA) ürününün hangi sürümü WebSphere Application Server ile birlikte teslim edilir?.

WebSphere Application Server Liberty and the IBM WebSphere MQ resource adapter
IBM WebSphere MQ Version 7.5 kaynak bağdaştırıcısı, wmqJmsClient-1.1 özelliği kullanılarak WebSphere Application Server Liberty Sürüm 8.5.5, Düzeltme Paketi 2 ya da sonraki bir sürüme kurulabilir. Diğer bir seçenek olarak, bazı kısıtlamalara tabi olarak, soysal Java Platform, Enterprise Edition Connector Architecture (Java EE JCA) desteği kullanarak kaynak bağdaştırıcısını kurabilirsiniz.

Kaynak bağdaştırıcısını Liberty 'ye kurarken genel sınırlamalar

wmqJmsClient-1.1 özelliği kullanılırken ve soysal JCA desteği kullanılırken Version 7.5 kaynak bağdaştırıcısı için aşağıdaki kısıtlamalar geçerli olur:

- IBM WebSphere MQ classes for Java , Liberty 'de desteklenmiyor. Bunlar, IBM WebSphere MQ Liberty ileti sistemi özelliği ya da genel JCA desteğiyle kullanılmamalıdır. Daha fazla bilgi için bakınız: Using WebSphere MQ Java Interfaces in J2EE/JEE Environments.
- IBM WebSphere MQ kaynak bağdaştırıcısı, BINDINGS_THEN_CLIENT iletim tipine sahiptir. Bu iletim tipi, IBM WebSphere MQ Liberty ileti sistemi özelliği içinde desteklenmez.
- IBM WebSphere MQ Advanced Message Security (IBM WebSphere MQ AMS) özelliği, IBM WebSphere MQ Liberty ileti sistemi özelliğinde yer almaz.

IBM WebSphere MQ Version 7.5 kaynak bağdaştırıcısı, wmqJmsClient-2.0 özelliği ile kullanılamaz.

WebSphere MQ Resource Adapter ürününün yapılandırılması
WebSphere MQ kaynak bağdaştırıcısını yapılandırmak için çeşitli JCA kaynakları ve sistem özellikleri tanımlardınız.

JCA kaynaklarını aşağıdaki kategorilerde tanımlayın:

- Tanılama izleme düzeyi gibi kaynak bağdaştırıcısının genel özelliklerini temsil eden ResourceAdapter nesnesinin özellikleri. Bu özellikler “ResourceAdapter nesnesinin yapılandırılması” sayfa 707içinde açıklanmıştır.
- Bir ActivationSpec nesnesinin özellikleri, gelen iletişim için bir MDB ' nin nasıl etkinleştirildiğini belirler. Bu özellikler “Kaynak bağdaştırıcısının gelen iletişim için yapılandırılması” sayfa 709içinde açıklanmıştır.
- Uygulama sunucusunun giden iletişim için JMS ConnectionFactory nesnesi yaratmak için kullandığı ConnectionFactory nesnesinin özellikleri. Bu özellikler “Kaynak bağdaştırıcısının giden iletişim için yapılandırılması” sayfa 721içinde açıklanmıştır.
- Uygulama sunucusunun giden iletişim için bir JMS Kuyruğu nesnesi ya da JMS Konu nesnesi yaratmak için kullandığı, denetimli bir hedef nesnenin özellikleri. Bu özellikler “Kaynak bağdaştırıcısının giden iletişim için yapılandırılması” sayfa 721içinde de açıklanmıştır.

WebSphere MQ kaynak bağdaştırıcısı RAR dosyası, kaynak bağdaştırıcısına ilişkin konuşlandırma tanımlayıcısını içeren META-INF/ra.xmladlı bir dosya içerir. Bu konuşlandırma tanımlayıcısı, https://java.sun.com/xml/ns/j2ee/connector_1_5.xsd konumundaki XML şemasıyla tanımlanır ve kaynak bağdaştırıcısına ve sağladığı hizmetlere ilişkin bilgileri içerir. Bir uygulama sunucusu, kaynak bağdaştırıcısı için bir konuşlandırma planı da gerektirebilir. Bu devreye alma planı uygulama sunucusuna özgüdür.

Örneğin, WebSphere Application Server Community Edition , geronimo-ra.xmladlı bir konuşlandırma planı gerektirir.

Güvenli Yuva Arabirimi Katmanı (SSL) kullanıyorsanız, aşağıdaki örnekte olduğu gibi, anahtar deposu dosyası ve güvenli depo dosyasının JVM sistem özellikleri olarak yerlerini belirtin:

```
java ... -Djavax.net.ssl.keyStore=
key_store_location
         -Djavax.net.ssl.trustStore=trust_store_location
         -Djavax.net.ssl.keyStorePassword=key_store_password
```

Bu özellikler, bir ActivationSpec ya da ConnectionFactory nesnesinin özellikleri olamaz ve bir uygulama sunucusu için birden fazla anahtar deposu belirtemezsiniz. Özellikler tüm JVM için geçerlidir ve uygulama sunucusunda çalışmakta olan diğer uygulamalar SSL bağlantıları kullanıyorsa uygulama sunucusunu etkileyebilir. Uygulama sunucusu, bu özellikleri farklı değerlere de döndürebilir. JMS için WebSphere MQ sınıflarıyla SSL kullanımıyla ilgili daha fazla bilgi için bkz. “JMS için WebSphere MQ sınıflarıyla SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) kullanılıyor” sayfa 869.

An installation verification test (IVT) program is supplied with the WebSphere MQ resource adapter, but you must configure the resource adapter before you can run the program. IVT programını çalıştırmak için yapılandırmanız gerekenlerle ilgili bilgi almak için bkz. “ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750.

Kaynak bağdaştırıcısı günlükleri, uyarı ve hata iletileri, JMS için IBM WebSphere MQ sınıflarıyla aynı mekanizmayı kullanır; ayrıntılar için bkz. “Günlüğe kaydetme ve IBM WebSphere MQ classes for JMS” sayfa 764. WebSphere Application Serveriçin, bu iletiler otomatik olarak uygulama sunucusu çıkış günlüğüne yeniden yönlendirilir. WAS CE ve JBossgibi diğer uygulama sunucuları için bu, varsayılan olarak mqjms.logadlı bir dosyaya gider. Kaynak bağdaştırıcısını, uygulama sunucularınız standart çıkış günlüğüne uyarı iletileri olarak kaydetmek üzere yapılandırmak için, uygulama sunucunuz için aşağıdaki JVM sistem özelliğini ayarlayın:

```
-Dcom.ibm.msg.client.commonservices.log.outputName=mqjms.log,stdout
```

Bir JVM sistem özelliğinin ayarlamaya ilişkin ayrıntılar için uygulama sunucusu belgelerinize bakın.

ResourceAdapter nesnesinin yapılandırılması
ResourceAdapter nesnesi, WebSphere MQ kaynak bağdaştırıcısının genel özelliklerini sarsalıyor. Bu özellikleri, kaynak bağdaştırıcınız için gereken olanakları kullanarak tanımlayın.

ResourceAdapter nesnesi için iki özellik kümesi vardır:

- Tanılama izlemesi ile ilişkili özellikler
- Kaynak bağdaştırıcısı tarafından yönetilen bağlantı havuzuyla ilişkili özellikler

Bu özellikleri tanımlamanızın yolu, uygulama sunucunuz tarafından sağlanan yönetim arabirimlerine bağlıdır.

Tanılama izlemesiyle ilişkili özelliklerin tanımlamaya ilişkin ek bilgi edinmek için IBM WebSphere MQ kaynak bağdaştırıcısının izlenmesibaşlıklı konuya bakın.

Kaynak bağdaştırıcısı, MDBS ' ye ileti göndermek için kullanılan bir iç bağlantı havuzu iç bağlantı havuzunu yönetir. Çizelge 95 sayfa 707 Bağlantı havuzuyla ilişkili ResourceAdapter nesnesinin özelliklerini listeler.

Çizelge 95. Bağlantı havuzuyla ilişkili ResourceAdapter nesnesinin özellikleri

Özelliğin adı	Tip	Varsayılan değer	Tanım
maxConnections	Dizgi	50	Bir WebSphere MQ kuyruk yöneticisine ve konuşlandırılan en çok sayıda MDBs ile bağlantı sayısı üst sınırı.

Çizelge 95. Bağlantı havuzuyla ilişkili ResourceAdapter nesnesinin özellikleri (devamı var)			
Özelliğin adı	**Tip**	**Varsayılan değer**	**Tanım**
connectionConcurrency	Dizgi	1	Bir JMS bağlantısını paylaşabilmek için en çok MDBS ' nin sayısı. Paylaşım bağlantıları olanaklı değil ve bu özellik her zaman 1 değerine sahip olur.
reconnectionRetrySayı	Dizgi	5	Bir bağlantı başarısız olursa, kaynak bağdaştırıcısı tarafından WebSphere MQ kuyruk yöneticisine yeniden bağlanmak için yapılan deneme sayısı üst sınırı.
reconnectionRetryInterval	Dizgi	300 000	Kaynak bağdaştırıcısının bir WebSphere MQ kuyruk yöneticisine yeniden bağlanmayı denemeden önce bekleyeceği süre (milisaniye olarak).
startupRetrySayısı	Dizgi	0	Uygulama sunucusu başlatıldığında kuyruk yöneticisi çalışmıyorsa, başlatma sırasında bir MDB ' yi çalıştırmaya ve bağlamaya ilişkin varsayılan değer sayısı.
startupRetryAralığı	Dizgi	30 000	Başlatma bağlantısı girişimleri arasındaki varsayılan uyku süresi (milisaniye cinsinden).

Uygulama sunucusunda bir MDB konuşlandırıldığında, yeni bir JMS bağlantısı yaratılır ve kuyruk yöneticisiyle başlayan bir etkileşim, maxConnection özelliği tarafından belirtilen bağlantı sayısı üst sınırının aşılmaması koşuluyla başlar. Bu nedenle, MDBS sayısı üst sınırı bağlantı sayısı üst sınırına eşittir. Devreye alınan MDBs sayısı bu üst sınıra ulaşırsa, başka bir MDB ' yi devreye alma girişimi başarısız olur. Bir MDB durdurulduysa, bağlantısı başka bir MDB tarafından kullanılabilir.

Genel olarak, birçok MDB konuşlandırılacaksa, maxConnections özelliğinin değerini artırmanız gerekir.

The reconnectionRetryCount and reconnectionRetryInterval properties govern the behavior of the resource adapter when connections to a WebSphere MQ queue manager fail, because of a network failure for example. When a connection fails, the resource adapter suspends the delivery of messages to all MDBs supplied by that connection for an interval specified by the reconnectionRetryInterval property. Bundan sonra, kaynak bağdaştırıcısı kuyruk yöneticisine yeniden bağlanmayı dener. Girişim başarısız olursa, kaynak bağdaştırıcısı, reconnectionRetryInterval özelliği tarafından belirtilen aralıklarla yeniden bağlanma girişiminde bulunulursa, reconnectionRetrySayı özelliği tarafından belirtilen sınıra ulaşılıncaya kadar. Tüm denemeler başarısız olursa, MSB ' ler el ile yeniden başlatılıncaya kadar teslim kalıcı olarak durdurulur.

Genel olarak, ResourceAdapter nesnesi denetim gerektirmez. Ancak, UNIX and Linux sistemlerinde tanılama izlemenin geçerli kılınmasını sağlamak için aşağıdaki özellikleri ayarlayabilirsiniz:

```
traceEnabled:       true
traceLevel:         10
```

Kaynak bağdaştırıcısı başlatılmamışsa, örneğin, WebSphere MQ kaynaklarını kullanan uygulamalar yalnızca istemci kapsayıcısında çalıştırılıyorsa, bu özellikler, kaynak bağdaştırıcısı başlatılmamışsa herhangi bir etkiye sahip değildir. Bu durumda, tanılama izlemesi için özellikleri Java Virtual Machine (JVM) sistem özellikleri olarak ayarlayabilirsiniz. Aşağıdaki örnekte gösterildiği gibi, özellikleri **java** komutundaki -D işaretini kullanarak ayarlayabilirsiniz:

```
java ... -DtraceEnabled=true -DtraceLevel=6
```

ResourceAdapter nesnesinin tüm özelliklerini tanımlamanıza gerek yoktur. Belirtilmeyen tüm özellikler varsayılan değerlerini alır. Yönetilen bir ortamda, özellikleri belirtmenin iki yolunu karıştırmamak daha iyi

olur. Bunları karıştırsanız, JVM sistem özellikleri ResourceAdapter nesnesinin özelliklerine göre öncelik kazanır.

Kaynak bağdaştırıcısının gelen iletişim için yapılandırılması
Gelen iletişimi yapılandırmak için bir ya da daha çok ActivationSpec nesnesinin özelliklerini tanımlayın.

ActivationSpec nesnesinin özellikleri, bir ileti sürücüsü Bean 'in (MDB) bir WebSphere MQ kuyruğundan JMS iletilerini nasıl aldığını belirler. MDB ' nin işlem davranışı, konuşlandırma tanımlayıcısında tanımlanır.

ActivationSpec nesnesi iki özellik kümesine sahiptir:

- WebSphere MQ kuyruk yöneticisine JMS bağlantısı yaratmak için kullanılan özellikler
- Belirli bir kuyruğa geldiklerinde iletileri zamanuyumsuz olarak teslim eden bir JMS bağlantısı tüketicisi oluşturmak için kullanılan özellikler

ActivationSpec nesnesinin özelliklerini tanımlama şekliniz, uygulama sunucunuz tarafından sağlanan denetim arabirimlerine bağlıdır.

Çizelge 96 sayfa 709 içinde, bir WebSphere MQ kuyruk yöneticisine JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri listelenir.

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri

Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
applicationName	Dizgi	• Çağıran sınıf adı (varsa), 28 karakterden uzun olmayacak şekilde ayarlanır. Yoksa, `WebSphere MQ Client for Java` dizgisi kullanılır.	Bir uygulamanın kuyruk yöneticisine kayıtlı olduğu ad. Bu uygulama adı, **DISPLAY CONN MQSC/PCF** komutuyla (alanın **APPLTAG**olarak adlandırıldığı) ya da IBM WebSphere MQ Gezgin **Uygulama Bağlantıları** görüntüsünde (burada alan **App name**olarak adlandırılır) gösterilir.
brokerCCDurSubQueue[1]	Dizgi	• **SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE** • Kuyruk adı	Bağlantı tüketicisinin sürekli abonelik iletileri aldığı kuyruğun adı
brokerCCSubKuyruk[1]	Dizgi	• **SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE** • Kuyruk adı	Bağlantı tüketicisinin sürekli olmayan abonelik iletilerini aldığı kuyruğun adı
brokerControlKuyruk[1]	Dizgi	• **SYSTEM.BROKER.CONTROL.QUEUE** • Kuyruk adı	Aracı denetim kuyruğunun adı
brokerQueueManager[1]	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Aracının çalıştığı kuyruk yöneticisinin adı
brokerSubQueue[1]	Dizgi	• **SYSTEM.JMS.ND.SUBSCRIBER.QUEUE** • Kuyruk adı	Sürekli olmayan bir ileti tüketicisinin iletileri aldığı kuyruğun adı

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
brokerVersion[1]	Dizgi	• **belirlenmedi** -Aracı V6 'dan V7' ye geçirildikten sonra, bu özelliği RFH2 üstbilgilerinin artık kullanılmayacak şekilde ayarlayın. Geçişten sonra bu özellik artık ilgili değildir. • **V1** - WebSphere MQ yayınlama/abone olma aracısını kullanmak için. Ya da uyumluluk kipinde bir WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ya da WebSphere Business Integration Message Broker aracısını kullanmak için. TRANSPORT, BIND ya da CLIENT olarak ayarlandıysa, bu değer varsayılan değerdir. • **V2** -Bir WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ya da WebSphere Business Integration Message Broker aracısını yerli kipte kullanmak için. TRANSPORT değeri DIRECT ya da DIRECTHTTP olarak ayarlandıysa, bu değer varsayılan değerdir.	Kullanılmakta olan aracının sürümü
ccdtURL	Dizgi	• **boş değerli** • Birörnek kaynak konum belirleyici (URL)	İstemci kanal tanımlama çizelgesini (CCDT) içeren dosyanın adını ve yerini tanımlayan ve dosyaya nasıl erişilebileceğini belirten URL
CCSID	Dizgi	• **819** • Java sanal makinesi (JVM) tarafından desteklenen kodlanmış karakter takımı tanıtıcısı	Bağlantıya ilişkin kodlanmış karakter takımı tanıtıcısı
kanal	Dizgi	• **SYSTEM.DEF.SVRCONN** • MQI kanalının adı	Kullanılacak MQI kanalının adı
cleanupInterval[1] (Temizleme Aralığı)	int	• **3600 000** • Artı bir tamsayı	Yayınlama/abone olma temizleme yardımcı programının arka plan çalıştırmaları arasındaki milisaniye cinsinden aralık
cleanupLevel[1]	Dizgi	• **GüVENLI** • YOK • güçlü • ZORLA • NONDUR	Aracıya dayalı abonelik deposu için temizleme düzeyi

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
clientID	Dizgi	• **boş değerli** • Bir istemci tanıtıcısı	Bağlantıya ilişkin istemci tanıtıcısı
cloneSupport	Dizgi	• **DEVRE Dışı** -Bir kerede tek bir sürekli konu abonesi çalıştırılabilir. • ENABLED-Aynı sürekli konu abonesinin iki ya da daha fazla eşgörünümü aynı anda çalışabilir, ancak her yönetim ortamının ayrı bir Java sanal makinesinde (JVM) çalışması gerekir.	Aynı sürekli konu abonesinin iki ya da daha fazla eşgörünümünün aynı anda çalışıp çalışamayacağını belirler
connectionNameListesi	Dizgi	• **localhost (1414)** • Her öğenin biçimi aldığı, virgüllerle ayrılmış öğelerden oluşan bir dizgi: `HOSTNAME(PORT)` Burada *HOSTNAME* , bir DNS adı ya da bir IP adresidir.	Gelen iletişim için kullanılan TCP/IP bağlantı adlarının listesi. Belirtildiğinde, **connectionNameList** **hostname** ve **port** özelliklerinin yerini alır. Bu özellik, çok eşgörünümlü kuyruk yöneticilerine yeniden bağlanmak için kullanılır. **connectionNameList** , formda **localAddress**ile benzer, ancak bununla karıştırılmamalıdır. **localAddress** , yerel iletişimin özelliklerini belirtirken, **connectionNameList** uzak kuyruk yöneticisine nasıl erişileceğini belirtir.
failIfSusturma	Boole	• **doğru** • yanlış	Kuyruk yöneticisi susturma durumundaysa, belirli yöntemlere yönelik çağrıların başarısız olup olmayacağını belirler
headerCompression	Dizgi	• **YOK** • SYSTEM-RLE ileti üstbilgisi sıkıştırması gerçekleştirilir	Bir bağlantıdaki üstbilgi verilerini sıkıştırmak için kullanılabilecek tekniklerin listesi

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
hostName	Dizgi	• **localhost** • Anasistem adı • Bir IP adresi	Kuyruk yöneticisinin bulunduğu sistemin anasistem adı ya da IP adresi. **hostname** ve **port** özellikleri, belirtildiğinde **connectionNameList** özelliği tarafından değiştirilir.
localAddress	Dizgi	• **boş değerli** • Şu biçimde bir dizgi: `[host_name][(low_port[,high_port])]` Burada *anasistem_adı* bir anasistem adı ya da IP adresi, *alt_kapı* ve *yüksek_kapı* TCP kapı numaralarıdır ve köşeli ayraçlar isteğe bağlı bir bileşeni gösterir	Bir kuyruk yöneticisine yönelik bağlantı için, bu özellik aşağıdakilerden birini ya da her ikisini de belirtir: • Kullanılacak yerel ağ arabirimi • Kullanılacak yerel kapı ya da yerel kapı aralığı **localAddress** , formda **connectionNameList**ile benzer, ancak bununla karıştırılmamalıdır. **localAddress** , yerel iletişimin özelliklerini belirtirken, **connectionNameList** uzak kuyruk yöneticisine nasıl erişileceğini belirtir.
messageCompression	Dizgi	• **YOK** • Boş karakterlerle ayrılmış olarak aşağıdaki değerlerden birinin ya da daha fazlasının listesi: RLE ZLIBFAST ZLIBHIGH	Bir bağlantıdaki ileti verilerini sıkıştırmada kullanılabilecek tekniklerin listesi
messageRetention[1]	Boole	• **true** -İstenmeyen iletiler giriş kuyruğunda kalır • false-İstenmeyen iletiler, yok etme seçeneklerine göre ele alınmaz	Bağlantı tüketicisinin giriş kuyruğunda istenmeyen iletileri alıkoyup saklamadığını belirler
messageSelection[1]	Dizgi	• **İstemci** • Aracı	İleti seçiminin JMS için WebSphere MQ sınıfları tarafından mı, yoksa aracı tarafından mı yapılacağını belirler. brokerVersion 1 değerine sahip olduğunda aracı tarafından ileti seçimi desteklenmez.

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
parola	Dizgi	• **boş değerli** • Parola	Kuyruk yöneticisine bağlantı yaratılırken kullanılacak varsayılan parola
yoklama aralığı[1]	int	• **5000** • Pozitif bir tamsayı	Bir oturumdaki her ileti dinleyicinin kuyruğunda uygun bir ileti yoksa, bu değer, her ileti dinleyicisinin kuyruğundan bir ileti almaya çalışmasından önce geçmesi gereken milisaniye cinsinden aralık üst sınırıdır. Sık sık oturumdaki ileti dinleyicilerden herhangi biri için uygun bir ileti yoksa, bu özelliğin değerini artırmayı düşünebilirsiniz. Bu özellik yalnızca TRANSPORT, BIND ya da CLIENT değerine sahipse geçerlidir.
kapı	int	• **1414** • TCP kapı numarası	Kuyruk yöneticisinin dinlediği kapı. **hostname** ve **port** özellikleri, belirtildiğinde **connectionNameList** özelliği tarafından değiştirilir.
providerVersion	dizgi	• **belirtilmedi** • Aşağıdaki biçimlerden birindeki bir dizgi – V.R.M.F – V.R.M – V.R – V Burada V, R, M ve F sıfırdan büyük ya da sıfıra eşit tamsayı değerleridir.	MDB ' nin bağlanmak istediği kuyruk yöneticisinin sürümü, yayını, değişiklik düzeyi ve düzeltme paketi.
queueManager	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Bağlanılacak kuyruk yöneticisinin adı
receiveExit [3]	Dizgi	• **boş değerli** • Virgülle ayrılmış bir ya da daha çok öğeden oluşan bir dizgi; burada her öğe, Java arabirimi için WebSphere MQ sınıflarını gerçekleştiren bir sınıfın tam olarak nitelenmiş adıdır, MQReceiveExit	Bir kanal alma çıkış programını ya da art arda çalıştırılacak bir alma çıkış programı sırasını tanımlar

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
receiveExitBaşlatma	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha fazla kullanıcı verisi öğesini içeren bir dizgi	Kanala geçirilen kullanıcı verileri, çağrıldığında çıkış programlarını alır
rescanInterval[1]	int	• **5000** • Pozitif bir tamsayı	Noktadan noktaya iletişim etki alanındaki bir ileti tüketicisi hangi iletileri almak istediğini seçmek için bir ileti seçici kullandığında, JMS için WebSphere MQ sınıfları WebSphere MQ kuyruğunda kuyruğun *MsgDeliverySequence* özniteliğinin belirlediği sırada uygun iletileri arar. JMS için WebSphere MQ sınıfları uygun bir ileti bulduğunda ve bunu tüketiciye teslim ettiğinde, JMS için WebSphere MQ sınıfları kuyruktaki yürürlükteki konumundan sonraki uygun iletiyi aramaya devam eder. WebSphere MQ JMS sınıfları, kuyruğun sonuna ulaşıncaya kadar ya da bu özelliğin değerinin belirlediği süre (milisaniye) doluncaya kadar kuyrukta arama yapmaya devam eder. Her bir durumda, JMS için WebSphere MQ sınıfları, aramaya devam etmek için kuyruğun başlangıcına döner ve yeni bir zaman aralığı başlar.
securityExit [3]	Dizgi	• **boş değerli** • Java arabirimi için WebSphere MQ sınıflarını gerçekleştiren bir sınıfın tam olarak nitelenmiş adı, MQSecurityExit	Kanal güvenliği çıkış programını tanımlar
securityExitBaşlatma	Dizgi	• **boş değerli** • Kullanıcı verileri dizgisi	Çağrıldığında bir kanal güvenliği çıkış programına geçirilen kullanıcı verileri
sendExit [3]	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha çok öğeden oluşan bir dizgi; burada her öğe, Java arabirimi için WebSphere MQ sınıflarını gerçekleştiren bir sınıfın tam olarak nitelenmiş adıdır, MQSendExit	Kanal gönderme çıkış programını ya da art arda çalıştırılacak çıkış gönderme programlarını tanımlar

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
sendExitInit	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha fazla kullanıcı verisi öğesini içeren bir dizgi	Çağrıldığında çıkış programlarını kanala gönderen kullanıcı verileri
shareConvİzin Verilir	Boole	• NO-Bir istemci bağlantısı yuvasını paylaşamıyor. • **YES** (EVET)-Bir istemci bağlantısı yuvasını paylaşabilir.	Bir istemci bağlantısının, kanal tanımlamaları eşleşiyorsa, aynı süreçten aynı kuyruk yöneticisine diğer üst düzey JMS bağlantılarıyla yuvasını paylaşıp paylaşamayacağını belirler
sparseSubAbonelikler[1]	Boole	• **false** -Abonelikler sık sık eşleşen iletiler alır. • true-Abonelikler nadiren eşleşen iletiler alır. Bu değer, abonelik kuyruğunun göz atmak için açılabilmesini gerektirir.	TopicSubscriber nesnesinin ileti alma ilkesini denetler
sslCertMağazaları	Dizgi	• **boş değerli** • Boşluklarla ayrılmış bir ya da daha çok LDAP URL dizesi. Her LDAP URL adresi şu biçimdedir: `ldap://host_name[:port]` Burada *anasistem_adı* bir anasistem adı ya da IP adresidir, *kapı* bir TCP kapı numarasıdır ve köşeli parantezler isteğe bağlı bir bileşeni gösterir.	SSL bağlantısında kullanılmak üzere sertifika iptal listelerini (CRL) tutan LDAP sunucuları
sslCipherÜrün Grubu	Dizgi	• **boş değerli** • CipherSuite adı	SSL bağlantısı için kullanılacak CipherSuite
sslFipsGerekli[2]	Boole	• **yanlış** • doğru	Bir SSL bağlantısının IBM Java JSSE FIPS sağlayıcısı (IBMJSSEFIPS) tarafından desteklenen bir CipherSuite kullanması gerekip gerekmediği
sslPeerAdı	Dizgi	• **boş değerli** • Ayırt edici adlar için şablon	Bir SSL bağlantısı için, kuyruk yöneticisi tarafından sağlanan sayısal sertifikada ayırt edici adı denetlemek için kullanılan bir şablon
sslResetSayısı	int	• **ERROR! SEGMENT DATA CORRUPTED, SEGDATA=0** • 0-999 999 999 aralığında bir tamsayı	SSL tarafından kullanılan gizli anahtarlar yeniden anlaşılmadan önce bir SSL bağlantısı tarafından gönderilen ve alınan toplam bayt sayısı

Çizelge 96. JMS bağlantısı yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
sslSocketÜreticisi	Dizgi	javax.net.ssl.SSLSocketFactory arabiriminin somutlamasını sağlayan bir sınıfın tam olarak nitelenmiş sınıf adını gösteren dizgi. İsteğe bağlı olarak, oluşturucu yöntemine geçirilecek ve ayraç içine alınmış bir bağımsız değişken de içerilir.	Yönetilen nesne kapsamında kurulan tüm bağlantılar, SSLSocketFactory arabiriminin bu somutlamasından elde edilen yuvaları kullanır.
statusRefreshInterval[1]	int	• **60000** • Pozitif bir tamsayı	Bir abonenin kuyruk yöneticisine olan bağlantısını kaybettiği zaman algılanan, uzun süren hareketin yenilenmesi arasındaki milisaniye cinsinden aralık. Bu özellik yalnızca subscriptionStore QUEUE değerine sahipse geçerlidir.
subscriptionStore[1]	Dizgi	• **Aracı** • GEÇİŞ YAPIN • kuyruk	JMS için WebSphere MQ sınıflarının etkin aboneliklerle ilgili kalıcı verileri nerede sakladığı belirler
transportType	Dizgi	• **İstemci** • Bağ Tanımları • BINDINGS_THEN_CLIENT	Bir kuyruk yöneticisine yönelik bağlantının istemci kipini mi, yoksa bağ tanımlama kipini mi kullandığını belirler. BINDINGS_THEN_CLIENT değeri belirtilirse, kaynak bağdaştırıcısı önce bağ tanımlama kipinde bağlantı kurmayı dener. Bu bağlantı girişimi başarısız olursa, kaynak bağdaştırıcısı istemci kipi bağlantısı kurmayı dener.
kullanıcı adı	Dizgi	• **boş değerli** • Kullanıcı adı	Bir kuyruk yöneticisine bağlantı yaratırken kullanılacak varsayılan kullanıcı adı
wildcardFormat	Dizgi	• CHAR-Aracı sürüm 1 'de kullanıldığı şekilde, yalnızca genel arama karakterlerini tanır • **TOPIC** -Aracı sürüm 2 'de kullanıldığı şekilde, yalnızca konu düzeyi genel arama karakterlerini tanır.	Genel arama karakteri sözdiziminin hangi sürümünün kullanılacağı

Notlar:

1. Bu özellik, JMS için WebSphere MQ sınıflarının 7.0 sürümüyle birlikte kullanılabilir. providerVersion özelliği 7 'den küçük bir sürüm numarasına ayarlanmadıkça, bu, bir Sürüm 7.0 kuyruk yöneticisine bağlı bir uygulamayı etkilemez.
2. sslFipsRequired özelliğini kullanma hakkında önemli bilgi için bkz. "IBM WebSphere MQ kaynak bağdaştırıcısına ilişkin sınırlamalar" sayfa 739.
3. Kaynak bağdaştırıcısının bir çıkış bulabilmesi için nasıl yapılandırılacağı hakkında bilgi için bkz. "Configuring IBM WebSphere MQ classes for JMS to use channel exits" sayfa 876.

Çizelge 97 sayfa 717 içinde, JMS bağlantı tüketicisi yaratmak için kullanılan bir ActivationSpec nesnesinin özellikleri listelenir.

Çizelge 97. JMS bağlantı tüketicisi yaratmak için kullanılan ActivationSpec nesnesinin özellikleri

Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
Hedef	Dizgi	Hedef adı	İletilerin alınacağı hedef. useJNDI özelliği, bu özelliğin değerinin nasıl yorumlanacağını belirler.
destinationType	Dizgi	• javax.jms.Queue • javax.jms.Topic	Hedef, kuyruk ya da konu tipi
maxMessages	int	• **1** • Artı bir tamsayı	Bir sunucu oturumuna aynı anda atanabilecek ileti sayısı üst sınırı. Etkinleştirme belirtimi bir XA hareketindeki bir MDB ' ye ileti sağlıyorsa, bu özelliğin ayarından bağımsız olarak 1 değeri kullanılır.
maxPoolDerinlik	int	• **10** • Artı bir tamsayı	Bağlantı tüketicisi tarafından kullanılan sunucu oturumu havuzundaki sunucu oturumu sayısı üst sınırı
messageSelector	Dizgi	• **boş değerli** • SQL92 ileti seçici ifadesi	Teslim edilecek iletileri belirten bir ileti seçici ifadesi
nonASFTimeout	int	• **ERROR! SEGMENT DATA CORRUPTED, SEGDATA=0** • Artı bir tamsayı	Pozitif bir değer, ASF dışı teslimatın kullanıldığını gösterir. Değer, bir alma isteğinin henüz ulaşmamış olabilecek iletileri (bekleme çağrısı ile alma) bekleyeceği süreyi milisaniye cinsinden belirtir. Varsayılan değer olan 0, ASF tesliminin kullanıldığını gösterir. Bu parametre yalnızca uygulama WebSphere Application Server sürüm 7 ya da sonraki bir yayın düzeyinde çalışıyorsa geçerlidir.
nonASFRollbackEtkin	Boole	• **false** -MDB başarısız olsa da ileti tüketilir. • true-MDB içindeki hata, iletinin kuyruğa geri işlenmesine neden olur.	MDB işlem yapılmamışsa, ileti teslimatının bir WebSphere MQ eşitleme noktası içinde olup olmadığını belirler. MDB işlemi yapıldıysa ya da nonASFTimeout değeri 0 olarak ayarlandıysa yoksayılır.

Çizelge 97. JMS bağlantı tüketicisi yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
poolTimeout	int	• **300000** • Artı bir tamsayı	Kullanılmayan bir sunucu oturumunun, etkinlik dışı durum nedeniyle kapatılmadan önce sunucu oturumu havuzunda açık tutulduğu süre (milisaniye)
readAheadİzin verilir	int	• **DESTINATION** -Kuyruk ya da konu tanımlamasına başvurarak önden okumaya izin verilip verilmediğini belirleyin. • DISABLED-Önden okumaya izin verilmez. • ENABLED-Önden okumaya izin verilir. • QUEUE-Kuyruk tanımına başvurarak önden okumaya izin verilip verilmediğini belirleyin. • KONU-Konu tanımlamasına başvurarak önden okumaya izin verilip verilmediğini belirleyin.	MDB ' nin hedeften kalıcı olmayan iletileri almadan önce bir iç arabelleğe almak için önden okuma özelliğini kullanıp kullanmasına izin verilip verilmediğini belirler
readAheadClosePolicy	int	• **ALL** -İç önden okuma arabelleğindeki tüm iletiler, durdurulmadan önce MDB ' ye teslim edilir. • CURRENT-İletileri daha sonra atılacak iç önden okuma arabelleğinde bırakarak, yalnızca yürürlükteki MDB çağrısı tamamlanır.	MDB yönetici tarafından durdurulduğunda iç önden okuma arabelleğindeki iletilere ne olur?
receiveCCSID	int	• **0** -JVM kullan `Charset.defaultCharset` • 1208- UTF-8 • Desteklenen bir kodlanmış karakter takımı tanıtıcısı	Kuyruk yöneticisi ileti dönüştürmesi için hedef CCSID ' yi belirleyen hedef özellik. **`receiveConversion`** , `QMGR` olarak ayarlanmadıkça değer yoksayılır.
receiveConversion	Dizgi	• **CLIENT_MSG** • QMGR	Veri dönüştürme işleminin kuyruk yöneticisi tarafından gerçekleştirilip gerçekleştirilmeyeceğini belirleyen hedef özellik.
startTimeout	int	• **10 000** • Artı bir tamsayı	İletiyi teslim etmek için çalışmanın zamanlanmasından sonra bir iletinin MDB ' ye tesliminin başlaması gereken süre (milisaniye). Bu süre geçerse, ileti kuyruğa geri alınır.

Çizelge 97. JMS bağlantı tüketicisi yaratmak için kullanılan ActivationSpec nesnesinin özellikleri (devamı var)			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
subscriptionDurability	Dizgi	• **NonDurable** -Sürekli olmayan abonelik, konuya abone olan bir MDB ' ye ileti teslim etmek için kullanılır. • Sürekli-Sürekli abonelik, konuya abone olan bir MDB ' ye ileti göndermek için kullanılır.	Konuya abone olan bir MDB ' ye ileti teslim etmek için sürekli ya da sürekli olmayan bir aboneliğin kullanılıp kullanılmayacağı
subscriptionName	Dizgi	• **"" (boş dizgi)** • Abonelik adı	Sürekli aboneliğin adı
useJNDI	Boole	• **false** -Hedef adı verilen özellik, bir WebSphere MQ kuyruğunun ya da konusunun adı olarak yorumlanır. • true-Hedef adı verilen özellik, uygulama sunucusunun JNDI ad alanında bir javax.jms.Queue nesnesinin ya da javax.jms.Topic nesnesinin adı olarak yorumlanır.	Hedef adı verilen özelliğin değerinin nasıl yorumlanacağını belirler

Hedef ve destinationType adı verilen ActivationSpec özellikleri belirtik olarak tanımlanmalıdır. Diğer tüm özellikler isteğe bağlıdır.

ActivationSpec nesnesi çakışan özelliklere sahip olabilir. Örneğin, bağ tanımlama kipinde bir bağlantı için SSL özelliklerini belirtebilirsiniz. Bu durumda davranış, destinationType özelliği tarafından belirlendiği şekilde, noktadan noktaya iletişim ya da yayınlama/abone olma olan iletim tipi ve ileti alışverişi etki alanı tarafından belirlenir. Belirtilen iletim tipi ya da ileti alışverişi etki alanı için geçerli olmayan özellikler yoksayılır.

Diğer özelliklerin tanımlanmasını gerektiren bir özellik tanımlarsanız, ancak bu diğer özellikleri tanımlamazsanız, ActivationSpec nesnesi bir MDB konuşlandırması sırasında validate () yöntemi çağrıldığında bir InvalidPropertykural dışı durumu verir. Kural dışı durum, uygulama sunucusunun yöneticisine uygulama sunucusuna bağlı olarak bildirilir. Örneğin, subscriptionDurability özelliğini Durable olarak ayarlarsanız, sürekli abonelikler kullanmak istediğinizi belirtirseniz, subscriptionName özelliğini de tanımlamanız gerekir.

ccdtURL adlı özellikler ve kanal tanımlandıysa, InvalidPropertykural dışı durumu yayınlanır. Ancak, yalnızca ccdtURL özelliğini tanımlarsanız, kanal adı verilen özelliği SYSTEM.DEF.SVRCONN, kural dışı durum yayınlanmaz ve bir JMS bağlantısını başlatmak için ccdtURL özelliğiyle tanıtılan istemci kanal tanımlama çizelgesi kullanılır.

ActivationSpec nesnesinin özelliklerinin çoğu, JMS nesneleri için WebSphere MQ sınıflarının özelliklerine ya da JMS yöntemleri için WebSphere MQ sınıflarının değiştirgelerine eşdeğerdir. Ancak, JMS için WebSphere MQ sınıflarında üç ayarlama özelliği ve bir kullanılabilirlik özelliği yoktur:

startTimeout
Kaynak bağdaştırıcısı bir İş nesnesini bir MDB ' ye ileti gönderecek şekilde zamanladıktan sonra, uygulama sunucusunun iş yöneticisinin kaynakların kullanılabilir olmasını bekleyeceği süre (milisaniye). İleti teslimi başlamadan önce bu süre geçerse, İş nesnesi zamanaşımına uğrarsa, ileti kuyruğa geri alınır ve kaynak bağdaştırıcısı iletiyi yeniden teslim etmeye çalışabilir. Etkinleştirildiyse, tanılama izlemesine bir uyarı yazılır, ancak ileti teslim etme işlemini başka bir şekilde etkilemez.

Bu koşulun yalnızca uygulama sunucusu çok yüksek bir yüke maruz kaldığında oluşmasını bekleyebilirsiniz. Koşul düzenli olarak oluşursa, iş yöneticisine ileti teslimini zamanlaması için daha uzun süre vermek üzere bu özelliğin değerini artırmayı düşünün.

maxPoolDerinlik
Bir bağlantı tüketicisi tarafından kullanılan sunucu oturumu havuzundaki sunucu oturumu sayısı üst sınırı. Bir sunucu oturumu yaratıldığında, bir kuyruk yöneticisiyle etkileşim başlatır. Bağlantı tüketicisi, bir iletiyi MDB ' ye teslim etmek için bir sunucu oturumu kullanır. Daha büyük bir havuz derinliği, yüksek hacimli durumlarda eşzamanlı olarak daha fazla ileti gönderilmesini sağlar, ancak uygulama sunucusunun daha fazla kaynağını kullanır. Birçok veritabanı konuşlandırılacaksa, uygulama sunucusundaki yükü yönetilebilir bir düzeyde tutmak için havuz derinliğini küçültmeyi düşünün. Her bağlantı tüketicisi kendi sunucu oturumu havuzunu kullanır; bu özellik, tüm bağlantı tüketicilerinin kullanabileceği toplam sunucu oturumu sayısını tanımlamaz.

poolTimeout
Kullanılmayan bir sunucu oturumunun, etkinlik dışı durum nedeniyle kapatılmadan önce sunucu oturumu havuzunda açık tutulduğu süre (milisaniye). İleti iş yükünde geçici bir artış, yükü dağıtmak için ek sunucu oturumlarının yaratılmasına neden olur, ancak ileti iş yükü normale döndükten sonra ek sunucu oturumları havuzda kalır ve kullanılmaz.

Bir sunucu oturumu her kullanıldığında, bir zaman damgasıyla işaretlenir. Belirli aralıklarla bir leştirici iş parçacığı, her sunucu oturumunun bu özellik tarafından belirtilen süre içinde kullanılıp kullanılmadığını denetler. Sunucu oturumu kullanılmamışsa, kapatılır ve sunucu oturumu havuzundan kaldırılır. Belirtilen süre geçtikten hemen sonra bir sunucu oturumu kapatılamayabilir; bu özellik, kaldırma işleminden önce etkinlik dışı kalma süresi alt sınırını gösterir.

useJNDI
Bu özelliğin açıklaması için bkz. Çizelge 97 sayfa 717.

Bir MDB 'yi konuşlandırmak için, önce MDB' nin gerektirdiği özellikleri belirterek ActivationSpec nesnesinin özelliklerini tanımlayın. Aşağıdaki örnek, belirtik olarak tanımlayabileceğiniz tipik bir özellik kümeidir:

```
channel:           SYSTEM.DEF.SVRCONN
destination:       SYSTEM.DEFAULT.LOCAL.QUEUE
destinationType:   javax.jms.Queue
hostName:          192.168.0.42
messageSelector:   color='red'
port:              1414
queueManager:      ExampleQM
transportType:     CLIENT
```

Uygulama sunucusu, daha sonra bir MDB ile ilişkilendirilmiş bir ActivationSpec nesnesi yaratmak için özellikleri kullanır. ActivationSpec nesnesinin özellikleri, iletilerin MDB ' ye nasıl teslim edileceğini belirler. MDB dağıtımlı hareketler gerektiriyorsa, ancak kaynak bağdaştırıcısı dağıtılmış hareketleri desteklemiyorsa MDB ' nin konuşlandırılması başarısız olur. Dağıtılmış hareketlerin desteklenmesi için kaynak bağdaştırıcısının nasıl kurulacağına ilişkin bilgi için bkz. “ WebSphere MQ Resource Adapter ürününün kuruluşu” sayfa 705.

Aynı hedeften birden çok MDB ileti alıyorsa, diğer MDB ' ler iletiyi almaya uygun olsa bile, noktadan noktaya iletişim etki alanında gönderilen bir ileti yalnızca bir MDB tarafından alınır. Özellikle, iki MDB farklı ileti seçiciler kullanıyorsa ve gelen bir ileti her iki ileti seçiciyle eşleşiyorsa, iletiyi yalnızca bir MDB alır. Bir iletiyi almak için seçilen MDB tanımlı değil ve iletiyi alan belirli bir MDB ' ye güvenemezsiniz. Yayınlama/abone olma etki alanında gönderilen iletiler, tüm uygun veritabanı yöneticileri tarafından alınır.

Kaynak Bağdaştırıcısında gelen zehirli ileti işleme

Bazı durumlarda, MDB ' ye teslim edilen bir ileti WebSphere MQ kuyruğuna geri işlenebilir. Bu geri alma işlemi, örneğin, daha sonra geriye işlenen bir iş birimi içinde bir ileti teslim edilirse gerçekleşebilir. Geriye işlenen bir ileti yeniden teslim edilir, ancak yanlış biçimlendirilmiş bir ileti defalarca bir MDB ' nin başarısız olmasına neden olabilir ve bu nedenle teslim edilemez. Böyle bir mesaj zehirli mesaj olarak adlandırılır. WebSphere MQ ' yu, JMS için WebSphere MQ sınıflarının daha ayrıntılı inceleme için başka bir kuyruğa otomatik olarak zehirli bir ileti aktarması ya da iletiyi atması için yapılandırabilirsiniz.

Zehirli iletilerin nasıl işleneceğine ilişkin ayrıntılar için bkz. “Handling poison messages in IBM WebSphere MQ classes for JMS” sayfa 853.

İlgili görevler

MQI istemcisinde çalıştırma zamanında yalnızca FIPS onaylı CipherSpecs kullanılmasının belirtilmesi

İlgili başvurular

UNIX, Linux ve Windows İçin Federal Bilgi İşleme Standartları (FIPS)

Kaynak bağdaştırıcısının giden iletişim için yapılandırılması

Giden iletişim yapılandırmak için bir ConnectionFactory nesnesinin özelliklerini ve denetlenen bir hedef nesneyi tanımlayın.

Giden iletişimi kullanırken, uygulama sunucusunda çalışan bir uygulama bir kuyruk yöneticisiyle bağlantı başlatır ve kuyruklarına ileti gönderir ve kuyruklarından iletileri zamanuyumlu olarak alır. Örneğin, doGet() sunucu uygulamacığı yöntemi giden iletişimi kullanır:

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
     throws ServletException, IOException {

...

// Look up ConnectionFactory and Queue objects from the JNDI namespace

     InitialContext ic = new InitialContext();
     ConnectionFactory cf = (javax.jms.ConnectionFactory) ic.lookup("myCF");
     Queue q = (javax.jms.Queue) ic.lookup("myQueue");

// Create and start a connection

     Connection c = cf.createConnection();
     c.start();

// Create a session and message producer

     Session s = c.createSession(false, Session.AUTO_ACKNOWLEDGE);
     MessageProducer pr = s.createProducer(q);

// Create and send a message

     Message m = s.createTextMessage("Hello, World!");
     pr.send(m);

// Create a message consumer and receive the message just sent

     MessageConsumer co = s.createConsumer(q);
     Message mr = co.receive(5000);

// Close the connection

      c.close();
}
```

Sunucu uygulamacığı bir HTTP GET isteği aldığında, JNDI ad alanından bir ConnectionFactory nesnesi ve bir kuyruk nesnesi alır ve bir WebSphere MQ kuyruğuna ileti göndermek için nesneleri kullanır. Daha sonra sunucu uygulaması gönderdiği iletiyi alır.

Giden iletişimini yapılandırmak için aşağıdaki kategorilerde JCA kaynakları tanımlayın:

- Uygulama sunucusunun bir JMS ConnectionFactory nesnesi yaratmak için kullandığı ConnectionFactory nesnesinin özellikleri.
- Uygulama sunucusunun JMS Kuyruğu nesnesi ya da JMS Konusu nesnesi yaratmak için kullandığı, denetlenen bir hedef nesnenin özellikleri.

Bu özellikleri tanımlama biçiminiz, uygulama sunucunuz tarafından sağlanan yönetim arabirimlerine bağlıdır. Uygulama sunucusu tarafından yaratılan ConnectionFactory, Kuyruk ve Konu nesneleri, bir uygulama tarafından alınabilecekleri bir JNDI ad alanına bağlanır.

Genellikle, uygulamaların bağlanması gerekebilecek her kuyruk yöneticisi için bir ConnectionFactory nesnesi tanımlarsınız. Uygulamaların noktadan noktaya iletişim etki alanında erişmesi gerekebilecek her kuyruk için bir Kuyruk nesnesi tanımlarsınız. Ve uygulamaların yayınlamak ya da abone olmak

isteyebileceği her konu için bir Konu nesnesi tanımlarsınız. ConnectionFactory nesnesi etki alanından bağımsız olabilir. Diğer bir seçenek olarak, noktadan noktaya iletişim etki alanı için etki alanına özgü bir QueueConnectionFactory nesnesi ya da yayınlama/abone olma etki alanı için bir TopicConnectionFactory nesnesi olabilir.

Çizelge 98 sayfa 722 içinde bir ConnectionFactory nesnesinin özellikleri listelenir.

Çizelge 98. ConnectionFactory nesnesinin özellikleri

Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
applicationName	Dizgi	• Çağıran sınıf adı (varsa), 28 karakterden uzun olmayacak şekilde ayarlanır. Yoksa, `WebSphere MQ Client for Java` dizgisi kullanılır.	Bir uygulamanın kuyruk yöneticisine kayıtlı olduğu ad. Bu uygulama adı, **`DISPLAY CONN MQSC/PCF`** komutuyla (alanın **`APPLTAG`**olarak adlandırıldığı) ya da IBM WebSphere MQ Gezgin **Uygulama Bağlantıları** görüntüsünde (burada alan **App name**olarak adlandırılır) gösterilir.
brokerCCSubKuyruk[1]	Dizgi	• **SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE** • Kuyruk adı	Bağlantı tüketicisinin sürekli olmayan abonelik iletileri aldığı kuyruğun adı.
brokerControlKuyruk[1]	Dizgi	• **SYSTEM.BROKER.CONTROL.QUEUE** • Kuyruk adı	Aracı denetim kuyruğunun adı.
brokerPubQueue[1]	Dizgi	• **SYSTEM.BROKER.DEFAULT.STREAM** • Kuyruk adı	Yayınlanan iletilerin gönderildiği kuyruğun adı (akış kuyruğu).
brokerQueueManager[1]	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Aracının çalıştığı kuyruk yöneticisinin adı.
brokerSubQueue[1]	Dizgi	• **SYSTEM.JMS.ND.SUBSCRIBER.QUEUE** • Kuyruk adı	Sürekli olmayan bir ileti tüketicisinin iletileri aldığı kuyruğun adı. Ek bilgi için BROKERSUBQ özelliğine bakın.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
brokerVersion[1]	Dizgi	• **belirlenmedi** -Aracı V6 'dan V7' ye geçirildikten sonra, bu özelliği RFH2 üstbilgilerinin artık kullanılmayacak şekilde ayarlayın. Geçişten sonra bu özellik artık ilgili değildir. • **V1** - IBM WebSphere MQ Yayınla/abone ol aracısını kullanmak için. Ya da uyumluluk kipinde bir IBM WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ya da WebSphere Business Integration Message Broker aracısını kullanmak için. TRANSPORT, BIND ya da CLIENT olarak ayarlandıysa, bu değer varsayılan değerdir. • **V2** -Bir IBM WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ya da WebSphere Business Integration Message Broker aracısını yerli kipte kullanmak için. TRANSPORT değeri DIRECT ya da DIRECTHTTP olarak ayarlandıysa, bu değer varsayılan değerdir.	Kullanılmakta olan aracının sürümü.
ccdtURL	Dizgi	• **boş değerli** • Birörnek kaynak konum belirleyici (URL)	İstemci kanal tanımlama çizelgesini (CCDT) içeren dosyanın adını ve yerini tanımlayan ve dosyaya nasıl erişilebileceğini belirten URL.
CCSID	Dizgi	• **819** • Java sanal makinesi (JVM) tarafından desteklenen bir kodlanmış karakter takımı tanıtıcısı	Bağlantıya ilişkin kodlanmış karakter takımı tanıtıcısı.
kanal	Dizgi	• **SYSTEM.DEF.SVRCONN** • MQI kanalının adı	Kullanılacak MQI kanalının adı.
cleanupInterval[1] (Temizleme Aralığı)	int	• **3600 000** • Artı bir tamsayı	Yayınlama/abone olma temizleme yardımcı programının arka plan çalıştırmaları arasındaki milisaniye cinsinden aralık.
cleanupLevel[1]	Dizgi	• **GüVENLI** • YOK • güçlü • ZORLA • NONDUR	Aracıya dayalı bir abonelik deposunun temizleme düzeyi.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
clientID	Dizgi	• **boş değerli** • Bir istemci tanıtıcısı	Bağlantıya ilişkin istemci tanıtıcısı.
cloneSupport	Dizgi	• **DEVRE Dışı** -Bir kerede tek bir sürekli konu abonesi çalıştırılabilir. • ENABLED-Aynı sürekli konu abonesinin iki ya da daha fazla eşgörünümü aynı anda çalışabilir, ancak her yönetim ortamının ayrı bir Java sanal makinesinde (JVM) çalışması gerekir.	Aynı sürekli konu abonesinin iki ya da daha fazla eşgörünümünün aynı anda çalışıp çalışamayacağını belirler.
connectionNameListesi	Dizgi	• **localhost (1414)** • Her öğenin biçimi aldığı, virgüllerle ayrılmış öğelerden oluşan bir dizgi: `HOSTNAME(PORT)` Burada *HOSTNAME* , bir DNS adı ya da bir IP adresidir.	Giden iletişim için kullanılan TCP/IP bağlantı adlarının listesi. **connectionNameList** , **hostname** ve **port** özelliklerinin yerini alır. Bu özellik, çok eşgörünümlü kuyruk yöneticilerine yeniden bağlanmak için kullanılır. **connectionNameList** , formda **localAddress**ile benzer, ancak bununla karıştırılmamalıdır. **localAddress** , yerel iletişimin özelliklerini belirtirken, **connectionNameList** uzak kuyruk yöneticisine nasıl erişileceğini belirtir.
failIfSusturma	Boole	• **doğru** • yanlış	Kuyruk yöneticisi susturma durumundaysa, belirli yöntemlere yönelik çağrıların başarısız olup olmayacağını belirler.
headerCompression	Dizgi	• **YOK** • SYSTEM-RLE ileti üstbilgisi sıkıştırması gerçekleştirilir.	Bir bağlantıdaki üstbilgi verilerinin sıkıştırılması için kullanılabilecek tekniklerin listesi.
hostName	Dizgi	• **localhost** • Anasistem adı • Bir IP adresi	Kuyruk yöneticisinin bulunduğu sistemin anasistem adı ya da IP adresi. **hostname** ve **port** özellikleri, belirtildiğinde **connectionNameList** özelliği tarafından değiştirilir.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama

localAddress	Dizgi	• **boş değerli**
		• Şu biçimde bir dizgi:
		`[host_name]`
		`[(low_port[,high_port])]`
		Burada *anasistem_adı* bir anasistem adı ya da IP adresi, *alt_kapı* ve *yüksek_kapı* TCP kapı numaralarıdır ve köşeli ayraçlar isteğe bağlı bir bileşeni gösterir
			Bir kuyruk yöneticisine yönelik bağlantı için, bu özellik aşağıdakilerden birini ya da her ikisini de belirtir:
			• Kullanılacak yerel ağ arabirimi
			• Kullanılacak yerel kapı ya da yerel kapı aralığı
			localAddress , formda **connectionNameList**ile benzer, ancak bununla karıştırılmamalıdır. **localAddress** , yerel iletişimin özelliklerini belirtirken, **connectionNameList** uzak kuyruk yöneticisine nasıl erişileceğini belirtir.

messageCompression	Dizgi	• **YOK** • Boş karakterlerle ayrılmış olarak aşağıdaki değerlerden birinin ya da daha fazlasının listesi: RLE ZLIBFAST ZLIBHIGH	Bir bağlantıdaki ileti verilerini sıkıştırmada kullanılabilecek tekniklerin listesi.
messageSelection[1]	Dizgi	• **İstemci** • Aracı	İleti seçiminin JMS için IBM WebSphere MQ sınıfları tarafından mı, yoksa aracı tarafından mı yapılacağını belirler. brokerVersion 1 değerine sahip olduğunda aracı tarafından ileti seçimi desteklenmez.
parola	Dizgi	• **boş değerli** • Parola	Kuyruk yöneticisine bağlantı yaratılırken kullanılacak varsayılan parola.
yoklama aralığı[1]	int	• **5000** • Pozitif bir tamsayı	Bir oturumdaki her ileti dinleyicinin kuyruğunda uygun bir ileti yoksa, bu değer, her ileti dinleyicisinin kuyruğundan bir ileti almaya çalışmasından önce geçmesi gereken milisaniye cinsinden aralık üst sınırıdır. Sık sık oturumdaki ileti dinleyicilerden herhangi biri için uygun bir ileti yoksa, bu özelliğin değerini artırmayı düşünebilirsiniz. Bu özellik yalnızca TRANSPORT, BIND ya da CLIENT değerine sahipse geçerlidir.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
kapı	int	• **1414** • TCP kapı numarası	Kuyruk yöneticisinin dinlediği kapı. **hostname** ve **port** özellikleri, belirtildiğinde **connectionNameList** özelliği tarafından değiştirilir.
providerVersion	dizgi	• **belirtilmedi** • Aşağıdaki biçimlerden birindeki bir dizgi – V.R.M.F – V.R.M – V.R – V Burada V, R, M ve F sıfırdan büyük ya da sıfıra eşit tamsayı değerleridir.	Uygulamanın bağlanmayı planladığı kuyruk yöneticisinin sürümü, yayın düzeyi, değişiklik düzeyi ve düzeltme paketi.
pubAckInterval[1]	int	• **25** • Artı bir tamsayı	IBM WebSphere MQ classes for JMS aracıdan bir alındı bildirimi istemeden önce bir yayınlayıcı tarafından yayınlanan iletilerin sayısı.
queueManager	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Bağlanılacak kuyruk yöneticisinin adı.
receiveExit[3]	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha çok öğeden oluşan bir dizgi; burada her öğe, IBM WebSphere MQ classes for Java arabirimini gerçekleştiren bir sınıfın tam olarak nitelenmiş adıdır (MQReceiveExit).	Bir kanal alma çıkış programını ya da arka arkaya çalıştırılacak bir alma çıkış programları sırasını tanımlar.
receiveExitBaşlatma	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha fazla kullanıcı verisi öğesini içeren bir dizgi	Kanala geçirilen kullanıcı verileri, çağrıldıklarında çıkış programlarını alır.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
rescanInterval[1]	int	• **5000** • Pozitif bir tamsayı	Noktadan noktaya iletişim etki alanındaki bir ileti tüketicisi, almak istediği iletileri seçmek için bir ileti seçici kullandığında, JMS için WebSphere MQ sınıfları IBM WebSphere MQ kuyruğunda kuyruğun *MsgDeliverySequence* özniteliğinin belirlediği sırada uygun iletileri arar. JMS için WebSphere MQ sınıfları uygun bir ileti bulduğunda ve bunu tüketiciye teslim ettiğinde, JMS için WebSphere MQ sınıfları kuyruktaki yürürlükteki konumundan sonraki uygun iletiyi aramaya devam eder. WebSphere MQ JMS sınıfları, kuyruğun sonuna ulaşıncaya kadar ya da bu özelliğin değerinin belirlediği süre (milisaniye) doluncaya kadar kuyrukta arama yapmaya devam eder. Her bir durumda, JMS için WebSphere MQ sınıfları, aramaya devam etmek için kuyruğun başlangıcına döner ve yeni bir zaman aralığı başlar.
securityExit[3]	Dizgi	• **boş değerli** • Java arabirimi için WebSphere MQ sınıflarını gerçekleştiren bir sınıfın tam olarak nitelenmiş adı, MQSecurityExit	Bir kanal güvenliği çıkış programını tanımlar.
securityExitBaşlatma	Dizgi	• **boş değerli** • Kullanıcı verileri dizgisi	Çağrıldığında kanal güvenliği çıkış programına geçirilen kullanıcı verileri.
sendCheckSayısı	int	• **ERROR! SEGMENT DATA CORRUPTED, SEGDATA=0** • Pozitif bir tamsayı	Hareket etmeyen tek bir JMS oturumunda, zamanuyumsuz koyma hatalarının denetlenmesi arasında izin verilecek gönderme çağrılarının sayısı.
sendExit[3]	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha çok öğeden oluşan bir dizgi; burada her öğe, Java arabirimi için WebSphere MQ sınıflarını gerçekleştiren bir sınıfın tam olarak nitelenmiş adıdır, MQSendExit	Kanal gönderme çıkış programını ya da art arda çalıştırılacak çıkış gönderme programlarını tanımlar.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
sendExitInit	Dizgi	• **boş değerli** • Virgüllerle ayrılmış bir ya da daha fazla kullanıcı verisi öğesini içeren bir dizgi	Kanala geçirilen kullanıcı verileri, çağrıldığında çıkış programlarını gönderir.
shareConvİzin Verilir	Boole	• NO-Bir istemci bağlantısı yuvasını paylaşamıyor. • **YES** (EVET)-Bir istemci bağlantısı yuvasını paylaşabilir.	Bir istemci bağlantısının, kanal tanımlamaları eşleşiyorsa, aynı süreçten aynı kuyruk yöneticisine diğer üst düzey JMS bağlantılarıyla yuvasını paylaşıp paylaşamayacağını belirler.
sparseSubAbonelikler[1]	Boole	• **false** -Abonelikler sık sık eşleşen iletiler alır. • true-Abonelikler nadiren eşleşen iletiler alır. Bu değer, abonelik kuyruğunun göz atmak için açılabilmesini gerektirir.	TopicSubscriber nesnesinin ileti alma ilkesini denetler.
sslCertMağazaları	Dizgi	• **boş değerli** • Boşluklarla ayrılmış bir ya da daha çok LDAP URL dizesi. Her LDAP URL adresi şu biçimdedir: `ldap://host_name[:port]` Burada *anasistem_adı* bir anasistem adı ya da IP adresidir, *kapı* bir TCP kapı numarasıdır ve köşeli parantezler isteğe bağlı bir bileşeni gösterir.	SSL bağlantısında kullanılmak üzere sertifika iptali listelerini (CRL ' ler) tutan LDAP sunucuları.
sslCipherÜrün Grubu	Dizgi	• **boş değerli** • CipherSuite adı	SSL bağlantısı için kullanılacak CipherSuite .
sslFipsGerekli[2]	Boole	• **yanlış** • doğru	Bir SSL bağlantısının, IBM Java JSSE FIPS sağlayıcısı (IBMJSSEFIPS) tarafından desteklenen bir CipherSuite kullanması gerekip gerekmediğini belirler.
sslPeerAdı	Dizgi	• **boş değerli** • Ayırt edici adlar için şablon	Bir SSL bağlantısı için, kuyruk yöneticisi tarafından sağlanan sayısal sertifikada ayırt edici adı denetlemek için kullanılan bir şablon.
sslResetSayısı	int	• **ERROR! SEGMENT DATA CORRUPTED, SEGDATA=0** • 0-999 999 999 aralığında bir tamsayı	SSL tarafından kullanılan gizli anahtarlar yeniden anlaşılmadan önce bir SSL bağlantısı tarafından gönderilen ve alınan toplam bayt sayısı.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
sslSocketÜreticisi	Dizgi	javax.net.ssl.SSLSocketFactory arabiriminin somutlamasını sağlayan bir sınıfın tam olarak nitelenmiş sınıf adını gösteren dizgi; isteğe bağlı olarak, oluşturucu yöntemine geçirilecek bir bağımsız değişken de içinde olmak üzere, ayraç içine alınmış olarak.	Yönetilen hedef nesne kapsamında kurulan tüm bağlantılar, SSLSocketFactory arabiriminin bu somutlamasından elde edilen yuvaları kullanır.
statusRefreshInterval[1]	int	• **60000** • Pozitif bir tamsayı	Bir abonenin kuyruk yöneticisine olan bağlantısını kaybettiği zaman algılanan, uzun süren hareketin yenilenmesi arasındaki milisaniye cinsinden aralık. Bu özellik yalnızca SUBSTORE ' un QUEUE değeri varsa anlamlıdır.
subscriptionStore[1]	Dizgi	• **Aracı** • GEÇİŞ YAPIN • kuyruk	JMS için WebSphere MQ sınıflarının etkin aboneliklerle ilgili kalıcı verileri nerede sakladığı belirler.
targetClientEşleştirme	Boole	• **doğru** • yanlış	Gelen bir iletinin JMSReplyTo üstbilgi alanıyla tanıtılan kuyruğa gönderilen bir yanıt iletisinin MQRFH2 üstbilgisinin yalnızca gelen iletinin MQRFH2 üstbilgisi varsa olup olmadığını belirler.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
temporaryModel	Dizgi	• **SYSTEM.DEFAULT.MODEL.QUEUE** • SYSTEM.JMS.TEMPQ.MODEL • Herhangi bir dizgi	JMS geçici kuyruklarının yaratıldığı model kuyruğunun adı. SYSTEM.DEFAULT.MODEL.QUEUE : • Uygulamanız, kalıcı olmayan iletileri kabul edecek geçici bir kuyruk kullanıyor. • Kuyruk yöneticisinde, ConnectionFactory ' in bir kerede gösterdiği geçici bir kuyruk yalnızca bir uygulama yaratır. SYSTEM.DEFAULT.MODEL.QUEUE , aynı anda yalnızca bir uygulama tarafından açılabilir. SYSTEM.JMS.TEMPQ.MODEL. Aşağıdaki durumlarda: • Uygulamanız kalıcı iletileri kabul edecek geçici bir kuyruk kullandığında. • Birden çok uygulama ConnectionFactory ' in gösterdiği kuyruk yöneticisine bağlanabiliyorsa ve bu uygulamaların aynı anda geçici kuyruklar oluşturmaları gerekiyorsa. DEFPSIST özniteliği YES olarak ayarlanmış yeni bir model kuyruğu tanımlayın ve aşağıdaki durumda DEFSOPT özniteliği SHARED olarak ayarlandı: • Uygulamanız kalıcı olmayan iletileri kabul edecek geçici bir kuyruk kullandığında ve birden çok uygulama ConnectionFactory ' in işaret ettiği kuyruk yöneticisine bağlandığında ve bu uygulamaların aynı anda geçici kuyruklar oluşturmaları gerekir. Yeni model kuyruğu yaratıldığında, temporaryModel özelliğini yeni model kuyruğunun adına ayarlayın.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
tempQPrefix	Dizgi	• **"" (boş dizgi)** • Bir IBM WebSphere MQ dinamik kuyruğunun adını oluşturmak için kullanılabilen bir önek. Öneki oluşturmaya ilişkin kurallar, IBM WebSphere MQ nesne tanımlayıcısı, MQOD yapısı içindeki *DynamicQName* alanının içeriğini oluşturmaya ilişkin kurallarla aynıdır, ancak son boş olmayan karakter bir yıldız işareti (*) olmalıdır. Özelliğin değeri boş dizgiyse, JMS için WebSphere MQ sınıfları AMQ.* değerini kullanır. Dinamik bir kuyruk yaratırken.	Bir IBM WebSphere MQ dinamik kuyruğunun adını oluşturmak için kullanılan önek.
tempTopicÖneki	Dizgi	IBM WebSphere MQ konu dizesi için yalnızca geçerli karakterlerden oluşan boş olmayan herhangi bir dizgi	Geçici konular oluştururken JMS, "TEMP/*TEMPTOPICPREFIX*/*unique_id*" biçiminde bir konu dizgisi oluşturur ya da bu özellik varsayılan değerle bırakılırsa, yalnızca "TEMP/*unique_id*". Boş olmayan bir TEMPTOPICPREFIX değerinin belirtilmesi, bu bağlantı altında yaratılan geçici konulara aboneler için yönetilen kuyruklar yaratmak üzere belirli model kuyruklarının tanımlanmasına olanak sağlar.
transportType	Dizgi	• **İstemci** • Bağ Tanımları • BINDINGS_THEN_CLIENT	Bir kuyruk yöneticisine yönelik bağlantının istemci kipini mi, yoksa bağ tanımlama kipini mi kullandığını belirler. BINDINGS_THEN_CLIENT değeri belirtilirse, kaynak bağdaştırıcısı önce bağ tanımlama kipinde bağlantı kurmayı dener. Bu bağlantı girişimi başarısız olursa, kaynak bağdaştırıcısı istemci kipi bağlantısı kurmayı dener.
kullanıcı adı	Dizgi	• **boş değerli** • Kullanıcı adı	Bir kuyruk yöneticisine bağlantı yaratırken kullanılacak varsayılan kullanıcı adı.
wildcardFormat	int	• CHAR-Aracı sürüm 1 'de kullanıldığı şekilde, yalnızca genel arama karakterlerini tanır • TOPIC-Aracı sürüm 2 'de kullanıldığı şekilde, yalnızca konu düzeyi genel arama karakterlerini tanır	Genel arama karakteri sözdiziminin hangi sürümünün kullanılacağı.

Çizelge 98. ConnectionFactory nesnesinin özellikleri (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama

Notlar:

1. Bu özellik, IBM WebSphere MQ classes for JMS Sürüm 7.0 ile birlikte kullanılabilir; ancak, providerVersion özelliği 7 'den küçük bir sürüm numarasına ayarlanmadıkça, bir Sürüm 7.0 kuyruk yöneticisine bağlı bir uygulamayı etkilemez.
2. sslFipsRequired özelliğini kullanma hakkında önemli bilgi için bkz. "IBM WebSphere MQ kaynak bağdaştırıcısına ilişkin sınırlamalar" sayfa 739.
3. Kaynak bağdaştırıcısının bir çıkış bulabilmesi için nasıl yapılandırılacağı hakkında bilgi için bkz. "Configuring IBM WebSphere MQ classes for JMS to use channel exits" sayfa 876.

Aşağıdaki örnek, bir ConnectionFactory nesnesinin tipik bir özellik kümesini göstermektedir:

```
channel:        SYSTEM.DEF.SVRCONN
hostName:       192.168.0.42
port:           1414
queueManager:   ExampleQM
transportType:  CLIENT
```

Çizelge 99 sayfa 732 içinde bir Kuyruk nesnesi ve Konu nesnesi için ortak olan özellikler listelenir.

Çizelge 99. Bir Kuyruk nesnesi ve Konu nesnesi için ortak olan özellikler

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
CCSID	Dizgi	• **1208** • Java sanal makinesi (JVM) tarafından desteklenen bir kodlanmış karakter takımı tanıtıcısı	Hedefe ilişkin kodlanmış karakter takımı tanıtıcısı.
Kodlama	Dizgi	• **Yerel** • Üç karakterden oluşan bir dizgi: – İlk karakter ikili tamsayıların gösterimini belirtir: - *N* , normal kodlamayı gösterir. - *R* ters kodlamayı belirtir. – İkinci karakter paketlenmiş ondalık tamsayıların gösterimini belirtir: - *N* , normal kodlamayı gösterir. - *R* ters kodlamayı belirtir. – Üçüncü karakter, kayan noktalı sayıların gösterimini belirtir: - *N* standart IEEE kodlamasını belirtir. - *R* , ters IEEE kodlamasını belirtir. - *3* , zSeries kodlamasını belirtir. NATIVE, NNN dizesine eşdeğerdir.	Hedef için ikili tamsayıların, paketlenmiş ondalık tamsayıların ve kayan noktalı sayıların gösterimi.

Çizelge 99. Bir Kuyruk nesnesi ve Konu nesnesi için ortak olan özellikler (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
Son kullanma tarihi	Dizgi	• **APP** -İletinin süre bitimi, ileti üreticisi tarafından belirlenir. • UNLIM-Bir iletinin süresi hiçbir zaman dolmaz. • 0-Bir iletinin süresi hiçbir zaman dolmaz. • Bir iletinin milisaniye cinsinden süre bitimini gösteren pozitif bir tamsayı.	Hedefe gönderilen bir iletinin süre bitimi.
failIfSusturma	Dizgi	• **doğru** • yanlış	Kuyruk yöneticisi susturma durumundaysa, hedefe erişme girişiminin başarısız olup olmadığını belirler.
Kalıcılık	Dizgi	• **APP** -Bir iletinin sürekliliği, ileti üreticisi tarafından belirlenir. • QDEF-Bir iletinin kalıcılığını, WebSphere MQ kuyruğunun *DefPersistence* özniteliği belirler. • PERS-Bir ileti kalıcı. • Hayır-Bir ileti kalıcı değil. • HIGH-Bir iletinin kalıcı olarak saklanması, “JMS kalıcı iletileri” sayfa 868içindeki açıklamaya göre WebSphere MQ kuyruğunun *NonPersistentMessageClass* özniteliği tarafından belirlenir.	Hedefe gönderilen bir iletinin kalıcılığı.
öncelik	Dizgi	• **APP** -Bir iletinin önceliği, ileti üreticisi tarafından belirlenir. • QDEF-Bir iletinin önceliği, IBM WebSphere MQ kuyruğunun *DefPriority* özniteliğine göre belirlenir. • 0, en düşük öncelik, 9, en yüksek öncelik aralığında bir tamsayı.	Hedefe gönderilen bir iletinin önceliği.
putAsync' e izin verilir	Dizgi	• QUEUE-Kuyruk tanımına başvurarak zamanuyumsuz girişlere izin verilip verilmediğini saptayın. • KONU-Konu tanımlamasına başvurarak zamanuyumsuz girişlere izin verilip verilmediğini saptayın. • DESTINATION-Kuyruk ya da konu tanımlamasına başvurarak zamanuyumsuz girişlere izin verilip verilmediğini belirler. • DISABLED-Zamanuyumsuz girişlere izin verilmez. • ENABLED-Zamanuyumsuz girişlere izin verilir.	İleti üreticilerinin bu hedefe ileti göndermek için zamanuyumsuz girişler kullanmalarına izin verilip verilmediğini belirler.

Çizelge 99. Bir Kuyruk nesnesi ve Konu nesnesi için ortak olan özellikler (devamı var)			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
readAheadİzin verilir	int	• **DESTINATION** -Kuyruk ya da konu tanımlamasına başvurarak önden okumaya izin verilip verilmediğini belirleyin. • DISABLED-Önden okumaya izin verilmez. • ENABLED-Önden okumaya izin verilir. • QUEUE-Kuyruk tanımına başvurarak önden okumaya izin verilip verilmediğini belirleyin. • KONU-Konu tanımlamasına başvurarak önden okumaya izin verilip verilmediğini belirleyin.	İleti tüketicilerinin ve kuyruk tarayıcılarının, hedeften kalıcı olmayan iletileri almadan önce bir iç arabelleğe almak için önden okuma özelliğini kullanıp kullanmalarına izin verilip verilmediğini belirler.
receiveCCSID	int	• **0** -JVM kullan `Charset.defaultCharset` • 1208- UTF-8 • Desteklenen bir kodlanmış karakter takımı tanıtıcısı	Kuyruk yöneticisi ileti dönüştürmesi için hedef CCSID ' yi belirleyen hedef özellik. **`receiveConversion`** , `QMGR` olarak ayarlanmadıkça değer yoksayılır.
receiveConversion	Dizgi	• **CLIENT_MSG** • QMGR	Veri dönüştürme işleminin kuyruk yöneticisi tarafından gerçekleştirilip gerçekleştirilmeyeceğini belirleyen hedef özellik.
targetClient	Dizgi	• **JMS** -İletinin hedefi bir JMS uygulamasıdır. • MQ -Bir iletinin hedefi JMS olmayan bir IBM WebSphere MQ uygulamasıdır.	Hedefe gönderilen bir iletinin hedefinin bir JMS uygulaması olup olmadığını belirler. Hedefi JMS uygulaması olan bir ileti MQRFH2 üstbilgisi içeriyor.

Çizelge 100 sayfa 734 içinde bir Kuyruk nesnesine özgü özellikler listelenir.

Çizelge 100. Kuyruk nesnesine özgü özellikler			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
baseQueueManagerName	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Temel IBM WebSphere MQ kuyruğunun sahibi olan kuyruk yöneticisinin adı.
baseQueueAdı	Dizgi	• **"" (boş dizgi)** • Kuyruk adı	Temel IBM WebSphere MQ kuyruğunun adı.

Çizelge 101 sayfa 734 içinde, bir Konu nesnesine özgü özellikler listelenir.

Çizelge 101. Konu nesnesine özgü özellikler			
Özelliğin adı	**Tip**	**Geçerli değerler (koyu varsayılan değer)**	**Açıklama**
baseTopicAdı	Dizgi	• **"" (boş dizgi)** • Konu adı	Temeldeki konunun adı.

Çizelge 101. Konu nesnesine özgü özellikler (devamı var)

Özelliğin adı	Tip	Geçerli değerler (koyu varsayılan değer)	Açıklama
brokerCCDurSubQueue[1]	Dizgi	• **SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE** • Kuyruk adı	Bağlantı tüketicisinin sürekli abonelik iletileri aldığı kuyruğun adı.
brokerDurSubQueue[1]	Dizgi	• **SYSTEM.JMS.D.SUBSCRIBER.QUEUE** • Kuyruk adı	Sürekli bir konu abonesinin iletileri aldığı kuyruğun adı. Ek bilgi için WebSphere MQ Explorer belgelerinde BROKEDURRSUBQ özelliğine bakın.
brokerPubQueue[1]	Dizgi	• **Ayarlanmadı** • Kuyruk adı	Yayınlanan iletilerin gönderildiği kuyruğun adı (akış kuyruğu). Bu özelliğin değeri, ConnectionFactory nesnesinin brokerPubQueue özelliğinin değerini geçersiz kılar. Ancak, bu özelliğin değerini ayarlamazsanız, bunun yerine ConnectionFactory nesnesinin brokerPubQueue özelliğinin değeri kullanılır.
brokerPubQueueManager[1]	Dizgi	• **"" (boş dizgi)** • Kuyruk yöneticisi adı	Konu üzerinde yayınlanan iletilerin gönderildiği kuyruğun iyesi olan kuyruk yöneticisinin adı.
brokerVersion[1]	Dizgi	• **Ayarlanmadı** • 1 • 2	Kullanılmakta olan aracının sürümü. Bu özelliğin değeri, ConnectionFactory nesnesinin brokerVersion özelliğinin değerini geçersiz kılar. Ancak, bu özelliğin değerini ayarlamazsanız, bunun yerine ConnectionFactory nesnesinin brokerVersion özelliğinin değeri kullanılır.

Not:

1. Bu özellik, IBM WebSphere MQ classes for JMS Sürüm 7.0 ile birlikte kullanılabilir, ancak ConnectionFactory nesnesinin providerVersion özelliği 7 'den küçük bir sürüm numarasına ayarlanmadıkça, 7.0 kuyruk yöneticisine bağlı bir uygulamayı etkilemez.

Aşağıdaki örnek, bir Kuyruk nesnesine ilişkin özellikler kümesini göstermektedir:

```
expiry:                UNLIM
persistence:           QDEF
baseQueueManagerName:  ExampleQM
baseQueueName:         SYSTEM.JMS.TEMPQ.MODEL
```

Aşağıdaki örnekte, bir Konu nesnesine ilişkin özellikler kümesi gösterilmektedir:

```
expiry:                UNLIM
persistence:           NON
baseTopicName:         myTestTopic
```

İlgili görevler

MQI istemcisinde çalıştırma zamanında yalnızca FIPS onaylı CipherSpecs kullanılmasının belirtilmesi

İlgili başvurular

UNIX, Linux ve Windows İçin Federal Bilgi İşleme Standartları (FIPS)

V 7.5.0.9 *Bir etkinleştirme belirtimi için targetClientEşleştirme özelliğinin yapılandırılması*

İstek iletileri bir MQRFH2 üstbilgisi içermediğinde yanıt iletilerine bir MQRFH2 üstbilgisi eklenebilmesi için bir etkinleştirme belirtimine ilişkin **targetClientMatching** özelliğini yapılandırabilirsiniz. Bu, bir uygulamanın yanıt iletisi üzerinde tanımladığı ileti özelliklerinin, ileti gönderildiğinde içerileceği anlamına gelir.

Bu görev hakkında

İletiyle yönlendirilen bir Bean (MDB) uygulaması, bir IBM WebSphere MQ JCA kaynak bağdaştırıcısı etkinleştirme belirtimi aracılığıyla bir MQRFH2 üstbilgisi içermeyen iletileri tüketir ve daha sonra istek iletisinin `JMSReplyTo` alanından oluşturulan JMS Hedefine yanıt iletileri gönderirse, istek iletileri olmasa bile yanıt iletilerinin bir MQRFH2 üstbilgisi içermesi gerekir. Tersi durumda, uygulamanın bir yanıt iletisi üzerinde tanımladığı ileti özellikleri kaybedilir.

targetClientMatching özelliği, gelen bir iletinin JMSReplyTo üstbilgi alanı tarafından tanımlanan kuyruğa gönderilen bir yanıt iletisinin, yalnızca gelen iletinin bir MQRFH2 üstbilgisi varsa MQRFH2 üstbilgisine sahip olup olmadığını tanımlar. You can configure this property for an activation specification, in both WebSphere Application Server traditional and WebSphere Application Server Liberty.

targetClientMatching özelliğinin değerini `false`olarak ayarladıysanız, bir MQRFH2içermeyen bir istek iletisinin JMSReplyTo üstbilgisinden yaratılmış bir JMS hedefine gönderilen bir yanıt iletisine bir MQRFH2 üstbilgisi eklenebilir. Bunun nedeni, JMS Hedefindeki **targetClient** özelliğinin `0`değerine ayarlandığından, bu da iletilerin bir MQRFH2 üstbilgisi içermesi anlamına gelir. Giden iletide MQRFH2 üstbilgisinin varlığı, IBM WebSphere MQ kuyruğuna gönderildiğinde, ileti üzerinde kullanıcı tanımlı ileti özelliklerinin saklamaya izin verir.

targetClientMatching özelliği `true` değerine ayarlıysa ve bir istek iletisi bir MQRFH2 üstbilgisi içermiyorsa, yanıt iletisine bir MQRFH2 üstbilgisi eklenmez.

Yordam

- In WebSphere Application Server traditional, use the administration console to define the **targetClientMatching** property as a custom property on the IBM WebSphere MQ activation specification:
  a) Gezinme bölmesinde **Resources-> JMS-> Activation spesifikasyonları**seçeneğini tıklatın.
  b) Görüntülemek ya da değiştirmek istediğiniz etkinleştirme belirtiminin adını seçin.
  c) **Custom properties-> New** (Özel özellikler-> Yeni) öğelerini tıklatın ve yeni özel özelliğin ayrıntılarını girin.
  Set the name of the property to `targetClientMatching`, the type to `java.lang.Boolean` and the value to `false`.
- In WebSphere Application Server Liberty, specify the **targetClientMatching** property on the definition of an activation specification within the `server.xml`.
  Örneğin:

```
<jmsActivationSpec id="SimpleMDBApplication/SimpleEchoMDB/SimpleEchoMDB">
<properties.wmqJms destinationRef="MDBRequestQ"
queueManager="MY_QMGR" transportType="BINDINGS" targetClientMatching="false"/>
<authData password="********" user="tom"/>
</jmsActivationSpec>
```

İlgili kavramlar

“Bir JMS uygulamasında hedefler oluşturma” sayfa 843

Bir Java Adlandırma ve Dizin Arabirimi (JNDI) ad alanından yönetilen nesneler olarak hedefleri almak yerine, bir JMS uygulaması yürütme sırasında dinamik dinamik hedefler yaratmak için bir oturum kullanabilir. An application can use a uniform resource identifier (URI) to identify a WebSphere MQ queue or a topic and, optionally, to specify one or more properties of a Queue or Topic object.

"Kaynak bağdaştırıcısının giden iletişim için yapılandırılması" sayfa 721
Giden iletişim yapılandırmak için bir ConnectionFactory nesnesinin özelliklerini ve denetlenen bir hedef nesneyi tanımlayın.

ASF ve ASF dışı kip
Application Server Facilitis (ASF) kipi, WebSphere Application Server ' daki ileti dinleyici hizmetinin iletileri işlediği varsayılan yöntemdir.

İleti dinleyici hizmetinin iki işlem kipi vardır: Application Server Facilitis (ASF) ve Application Server Facilitial Facilitis (ASF olmayan):

- ASF kipi uygulamalar için eşzamanlılık ve işlemsel destek sağlar. Yayınlama/abone olma ileti-sürücü Bean 'leri için ASF kipi, ASF dışı kipte dinleyici tek iş parçacıklı olduğu için daha iyi iş oranı ve eşzamanlılık sağlar.
- ASF dışı kip daha çok, JMS belirtimine isteğe bağlı bir uzantı olan JMS ASF ' yi desteklemeyen üçüncü taraf ileti alışverişi sağlayıcılarıyla birlikte kullanılır. ASF dışı kip de işlemsel olmakla birlikte, yol uzunluğu ASF kipinden daha kısa olduğu için, genellikle iyileştirilmiş başarım sağlar.

Uygulama sunucusundaki tüm iletilerle yönlendirilen Bean dinleyicileri için ASF olmayan bir işlem kipini etkinleştirmek için bu özelliği sıfır olmayan bir değere ayarlayın.

Not:

Non-ASF mode cannot be selected on z/OS systems, so you must not set a non-zero value for this property in this case.

ASF kipindeki ileti işleme

ASF kipinde, sunucu oturumları ve iş parçacıkları yalnızca, ileti odaklı bean (MDB) için uygun bir ileti saptandığında iş için ayrılır. Bir MDB ' nin koşutzamanlı olarak işleyebileceği iş parçacıklarının sayısı, dinleyici kapısı ya da etkinleştirme belirtimi için **Maximum Sessions** özelliğinin değerine göre belirlenir.

ASF olmayan kipte ileti işleme

ASF olmayan kipte iş parçacıkları, dinleyici kapısı ya da etkinleştirme belirtiminin başlatıldığı andan itibaren etkindir. Etkin iş parçacıklarının sayısı, **Maximum Sessions** özelliği için belirtilen değer tarafından belirlenir. The number of threads specified in **Maximum Sessions** property are active, regardless of the number of messages that are available to be processed. Her etkin iş parçacığı, tek bir fiziksel ağ bağlantısıdır.

IBM WebSphere MQ Sürüm 7.0 ya da üstü, tek bir fiziksel ağ bağlantısını paylaşan en fazla on iş parçacığını bulunmanıza olanak tanır.

İlgili kavramlar
JMS Application Server Facilitis içinIBM WebSphere MQ sınıfları
Bu konuda, JMS için WebSphere MQ sınıflarının Oturum sınıfındaki ConnectionConsumer sınıfını ve gelişmiş işlevselliğini nasıl gerçekleştirdiğini ele alır. Ayrıca, bir sunucu oturum havuzunun işlevi de özetlenir.

İlgili görevler
ASF dışı kip için etkinleştirme belirtimlerinin yapılandırılması
Etkinleştirme belirtimleri, WebSphere Application Server içinde çalışan bir ileti odaklı bean (MDB) ile IBM WebSphere MQiçinde bir hedef arasındaki ilişkiyi yönetmek ve yapılandırmak için standartlaştırılmış bir yöntemdir. Bu kısımda, iletilerin işlenmek için ASF olmayan kiplerin kullanılması için WebSphere Application Server ' in nasıl yapılandırılacağı açıklanmaktadır.

İlgili bilgiler

ASF kipindeki ve ASF olmayan kipte ileti işleme

ASF dışı kip için etkinleştirme belirtimlerini yapılandırma

Etkinleştirme belirtimleri, WebSphere Application Server içinde çalışan bir ileti odaklı bean (MDB) ile IBM WebSphere MQiçinde bir hedef arasındaki ilişkiyi yönetmek ve yapılandırmak için standartlaştırılmış bir yöntemdir. Bu kısımda, iletilerin işlenmek için ASF olmayan kiplerin kullanılması için WebSphere Application Server ' in nasıl yapılandırılacağı açıklanmaktadır.

Başlamadan önce

Bir etkinleştirme belirtiminin özelliklerini tanımlamanızın yolu, uygulama sunucunuz tarafından sağlanan yönetim arabirimlerine bağlıdır. Bu görev, uygulama sunucunuz olarak WebSphere Application Server sürüm 7 ya da daha sonraki bir sürümü ve ileti sistemi sağlayıcınız olarak IBM WebSphere MQ sürümünü kullandığınızı varsayar.

Not:

Non-ASF mode cannot be selected on z/OS systems.

Bu görev hakkında

Bir etkinleştirme belirtiminin özellikleri, bir ileti sürücüsü Bean 'in (MDB) JMS iletilerini bir IBM WebSphere MQ kuyruğundan nasıl aldığını belirler. ASF olmayan kipi yapılandırmak için bir ya da daha çok etkinleştirme belirtimlerinin özelliklerini tanımlayın.

ASF dışı kipte kullanabileceğiniz birkaç IBM WebSphere MQ yapılandırması vardır. Aşağıdaki yapılandırmalarda her bir iş parçacığı ayrı bir fiziksel ağ bağlantısı kullanır:

- An IBM WebSphere MQ Version 7.x queue manager, using a connection factory that has the Provider version property set to 6.
- An IBM WebSphere MQ Version 7.x queue manager, using a connection factory that has the Provider version property set to 7 or unspecified, connecting over a IBM WebSphere MQ channel that has the **SHARECNV** (sharing conversations) parameter set to 0.

To configure non-ASF, set the ActivationSpec property **NON.ASF.RECEIVE.TIMEOUT** to a positive integer, that indicates that non-ASF delivery is used. Bu değer, alma isteğinin henüz ulaşmamış olabilecek iletileri (bekleme çağrısı ile alma) beklediği süredir (milisaniye cinsinden). Varsayılan değer olan 0, ASF tesliminin kullanıldığını gösterir. Daha fazla ayrıntı için bkz. **İleti dinleyici hizmeti özel özellikleri**.

Bu parametre yalnızca, uygulama WebSphere Application Server sürüm 7 ya da sonraki bir sürüm üzerinde çalışırken geçerlidir.

Yordam

1. WebSphere Application Server yönetim konsolunu başlatın.
2. Dinleyici hizmeti ayarları sayfasını görüntüle:
   a) Gezinme bölmesinde **Servers > Server Types > WebSphere application servers**(Sunucular > Sunucu Tipleri > WebSphereapplicationuygulama sunucuları)
   b) İçerik bölmesinde uygulama sunucusunun adını tıklatın.
   c) **Communications**(İletişim) altında **Messaging > Message Listener Service**öğesini tıklatın.
3. **NON.ASF.RECEIVE.TIMEOUT** özel özelliğini, ileti dinleyici hizmetinin Özel özellikleri olarak ayarlayın.
   a) **Özel özellikler**' i tıklatın.
   b) **Yeni**'yi tıklatın.
   c) Enter the name of the property, **NON.ASF.RECEIVE.TIMEOUT**, in the **Ad** field.
   d) Enter the value you require, in the **Değer** field.
   e) **Tamam**'ı tıklatın.

4. Yaptığınız değişiklikleri ana yapılanışınıza saklayın.
5. Değiştirilen yapılandırmayı etkinleştirmek için, uygulama sunucusunu durdurup yeniden başlatın.

Sonuçlar

WebSphere Application Server için ileti dinleyici hizmetinin özelliklerini ASF olmayan kipi kullanacak şekilde yapılandırdınız.

Not: ASF olmayan kipi kullanırken, istenmeyen işlem zamanaşımlarını önlemek için, toplam hareket süresi zamanaşımı süresine ulaşılmadan önce, işlemin tamamlanmasına yetecek miktarda süre izin verdiğinizden emin olmanız gerekir. Daha fazla ayrıntı için bkz. **NON.ASF.RECEIVE.TIMEOUT** (WebSphere Application Server ürün belgelerinde).

İlgili kavramlar

“ASF ve ASF dışı kip” sayfa 737
Application Server Facilitis (ASF) kipi, WebSphere Application Server ' daki ileti dinleyici hizmetinin iletileri işlediği varsayılan yöntemdir.

Kaynak bağdaştırıcısının gelen iletişim için yapılandırılması
Gelen iletişimi yapılandırmak için bir ya da daha çok ActivationSpec nesnesinin özelliklerini tanımlayın.

İlgili bilgiler

İletiyle yönlendirilen Bean 'ler
İleti dinleyici hizmeti
ASF kipindeki ve ASF olmayan kipte ileti işleme
ASF olmayan kipte iletilerin nasıl işlendiği

IBM WebSphere MQ kaynak bağdaştırıcısına ilişkin sınırlamalar

IBM WebSphere MQ kaynak bağdaştırıcısını kullanırken, IBM WebSphere MQ ' un bazı özellikleri kullanılamaz ya da sınırlandırılır.

IBM WebSphere MQ kaynak bağdaştırıcısı aşağıdaki sınırlamalara sahiptir:

- IBM WebSphere MQ kaynak bağdaştırıcısı, z/OSdışında tüm IBM WebSphere MQ platformlarında desteklenir.
- IBM WebSphere MQ kaynak bağdaştırıcısı bir aracıya gerçek zamanlı bağlantıları desteklemez. Yalnızca istemci ya da bağ tanımları kipinde bir IBM WebSphere MQ kuyruk yöneticisine yönelik bağlantıları destekler.
- IBM WebSphere MQ kaynak bağdaştırıcısı, Java dışında dillerde yazılmış kanal çıkış programlarını desteklemez.
- Bir uygulama sunucusu çalışırken, tüm JCA kaynakları için gerekli olan tüm JCA kaynakları için true (doğru) ya da tüm JCA (yanlış) kaynakları için false (doğru) değeri olmalıdır. JCA kaynakları koşutzamanlı olarak kullanılmasa da bu bir gereksinmedir. If the sslFipsRequired property has different values for different JCA resources, IBM WebSphere MQ issues the reason code MQRC_UNSUPPORTED_CIPHER_SUITE, even if an SSL connection is not being used.
- Bir uygulama sunucusu için birden çok anahtar deposu belirtemezsiniz. Bağlantılar birden çok kuyruk yöneticisinden yapılırsa, tüm bağlantıların aynı anahtar deposunu kullanması gerekir. Bu sınırlama, WebSphere Application Server için geçerli değildir.
- İstemci kanal tanımlama çizelgesi (CCDT), birden çok uygun istemci bağlantısı kanal tanımlamasıyla birlikte kullanırsanız, kaynak bağdaştırıcısının farklı bir kanal tanımlaması ve dolayısıyla CCDT ' den farklı bir kuyruk yöneticisi seçmesi durumunda, işlem kurtarma sorunlarına neden olabilir. Kaynak bağdaştırıcısı, böyle bir yapılandırmanın kullanılmasını önlemek için herhangi bir işlem yapmaz ve hareket kurtarmasına ilişkin sorunlara neden olabilecek yapılandırmalardan kaçınmak sizin sorumluluğunuzda olur.
- IBM WebSphere MQ Version 7.0.1 içinde tanıtılan bağlantı yeniden deneme işlevselliği, bir JEE taşıyıcısında (EJB/Servlet) çalışırken giden bağlantılar için desteklenmiyor. Bağdaştırıcı bir JEE kapsayıcısı bağlamında, hareket yapılandırmasından ya da iletilmeyen kullanımdan bağımsız olarak kullanıldığında, giden JMS için bağlantı yeniden deneme işlemi desteklenmez.

İlgili görevler

MQI istemcisinde çalıştırma sırasında yalnızca FIPS onaylı CipherSpecs ' in kullanıldığını belirtme

İlgili başvurular

UNIX, Linux ve Windows için Federal Bilgi İşleme Standartları (FIPS)

JMS uygulamaları için WebSphere MQ sınıflarına ilişkin kuruluş sonrası ayarları

This topic tells you what authorities WebSphere MQ classes for JMS applications need in order to access the resources of a queue manager. Ayrıca, bağlantı kipleri tanıtılır ve uygulamaların istemci kipinde bağlanabilmesi için kuyruk yöneticisinin nasıl yapılandırılacağı anlatılır.

WebSphere MQ benioku dosyasını kontrol etmeyi unutmayın. Bu konuyla ilgili bilgilerin yerini alan bilgiler içerebilir.

JMS tarafından, ayrıcalıklı olmayan kullanıcılar için yetkilendirme gerektiren nesneler

Ayrıcalıklı olmayan kullanıcıların, JMS tarafından kullanılan kuyruklara erişmek için yetkilendirmeye izin verilmesi gerekir. Her JMS uygulamasının, çalıştığı kuyruk yöneticisi için yetkilendirmesi gerekir.

IBM WebSphere MQ' ta erişim denetime ilişkin ayrıntılar için bkz. Pencereler, UNIX and Linux sistemleri üzerinde güvenliğin ayarlanması .

JMS uygulamalarına ilişkin WebSphere MQ sınıflarının kuyruk yöneticisi için `bağlantı` ve `inq` yetkisi gerekir. **`setmqaut`** denetim komutunu kullanarak uygun yetkiler ayarlayabilirsiniz. Örneğin:

```
setmqaut -m QM1 -t qmgr -g jmsappsgroup +connect +inq
```

Noktadan noktaya iletişim etki alanı için aşağıdaki yetkiler gereklidir:

- MessageProducer nesneleri tarafından kullanılan kuyruklar `put` yetkisine gereksinim duyarlar.
- MessageConsumer ve QueueBrowser nesnelerine göre kullanılan kuyruklar `get`, `inq`ve `browse` yetkilerine gereksinim duyarlar.
- QueueSession.createTemporaryQueue () yönteminin, QueueConnectionFactory nesnesinin TEMPMODEL özelliği tarafından belirlenen model kuyruğuna erişmesi gerekir. Varsayılan olarak bu model kuyruğu SYSTEM.TEMP.MODEL.QUEUE.

Bu kuyruklardan herhangi birinin diğer ad kuyrukları varsa, bunların hedef kuyrukları sorgu yetkisi gerektirir. Hedef kuyruk bir küme kuyruğiyse, aynı zamanda göz atma yetkisi de gerektirir.

Yayınlama/abone olma etki alanı için, JMS için WebSphere MQ sınıfları, IBM WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipinde bir IBM WebSphere MQ kuyruk yöneticisine bağlanıyorsa, aşağıdaki kuyruklar kullanılır:

- SYSTEM.JMS.ADMIN.QUEUE
- SYSTEM.JMS.REPORT.QUEUE
- SYSTEM.JMS.MODEL.QUEUE
- SYSTEM.JMS.PS.STATUS.QUEUE
- SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
- SYSTEM.JMS.D.SUBSCRIBER.QUEUE
- SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE
- SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE
- SYSTEM.BROKER.CONTROL.QUEUE

IBM WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipi hakkında daha fazla bilgi için bkz. **PROVIDERVERSION** ne zaman ne zaman kullanılmalı

Buna ek olarak, JMS için WebSphere MQ sınıfları bu kipte bir kuyruk yöneticisine bağlanıyorsa, iletilerin yayınlandığı uygulamanın TopicConnectionFactory ya da konu nesnesi tarafından belirlenen akış kuyruğuna erişmesi gerekir. Varsayılan değer olarak, bu kuyruk SYSTEM.BROKER.DEFAULT.STREAM.

ConnectionConsumer, IBM WebSphere MQ Resource Adapter ya da WebSphere Application Server IBM WebSphere MQ ileti alışverişi sağlayıcısı kullanıyorsanız, ek yetkilendirmeye gerek olabilir.

ConnectionConsumer tarafından okunalacak kuyrukların `get` , `inq`ve `browse` yetkilerine sahip olması gerekir. Sistemin öldüğü mektup kuyruğu ve ConnectionConsumer tarafından kullanılan geri yedekleme kuyruğunda ya da rapor kuyruğunun `tak` ve `düzgeçiş` yetkilerine sahip olması gerekir.

Bir uygulama, yayınlama/abone olma ileti alışverişi gerçekleştirmek için WebSphere MQ ileti alışverişi sağlayıcısı olağan kipini kullandığında, uygulama kuyruk yöneticisi tarafından sağlanan tümleşik yayınlama/abone olma işlevinden yararlanir. Kullanılan konuların ve kuyrukların güvenliğini sağlamaya ilişkin bilgi için bkz. Güvenliği yayınla/abone ol .

JMS için WebSphere MQ sınıflarına ilişkin bağlantı kipleri

JMS uygulaması için bir WebSphere MQ sınıfları, istemci ya da bağ tanımları kipinde bir kuyruk yöneticisine bağlanabiliyor. İstemci kipinde, JMS için WebSphere MQ sınıfları TCP/IP üzerinden kuyruk yöneticisine bağlanır. Bağ tanımları kipinde, JMS için WebSphere MQ sınıfları Java Native Interface (JNI) olanağını kullanarak doğrudan kuyruk yöneticisine bağlanır.

z/OS üzerinde WebSphere Application Server 'da çalışan bir uygulama, bağ tanımlarında ya da istemci kipinde bir kuyruk yöneticisine bağlanabilir, ancak z/OS üzerinde başka bir ortamda çalışan bir uygulama, kuyruk yöneticisine yalnızca bağ tanımları kipinde bağlanabilir. Başka bir altyapıda çalışan bir uygulama, bağ tanımlarında ya da istemci kipinde kuyruk yöneticisine bağlanabilir.

Geçerli bir kuyruk yöneticisiyle JMS için geçerli ya da daha önceki bir WebSphere MQ sınıfı sürümünü kullanabilir ve JMS için geçerli WebSphere MQ sınıflarının yürürlükteki sürümüyle, kuyruk yöneticisinin yürürlükteki ya da daha önceki bir sürümünü kullanabilirsiniz. Farklı sürümleri karıştırsanız, işlev daha önceki sürümün düzeyiyle sınırlıdır.

Aşağıdaki kısımlarda, bağlantı kiplerinin her biri daha ayrıntılı olarak açıklanmıştır.

İstemci kipi

İstemci kipinde bir kuyruk yöneticisine bağlanmak için, JMS uygulaması için bir WebSphere MQ sınıfları kuyruk yöneticisinin çalıştığı sistemde ya da farklı bir sistemde çalıştırılabilir. Her durumda, JMS için WebSphere MQ sınıfları TCP/IP üzerinden kuyruk yöneticisine bağlanır.

Bağ tanımları kipi

Bağ tanımları kipindeki bir kuyruk yöneticisine bağlanmak için, JMS uygulamasına ilişkin bir WebSphere MQ sınıflarının kuyruk yöneticisinin çalıştığı sistemde çalışması gerekir.

JMS için WebSphere MQ sınıfları, Java Native Interface (JNI) olanağını kullanarak doğrudan kuyruk yöneticisine bağlanır. Bağ tanımları aktarımında kullanmak için, JMS için WebSphere MQ sınıfları WebSphere MQ Java Native Interface kitaplıklarına erişimi olan bir ortamda çalıştırılmalıdır; ek bilgi için “Java Native Interface (JNI) kitaplıklarının yapılandırılması” sayfa 696 ' e bakın.

JMS için WebSphere MQ sınıfları, *ConnectOption* için aşağıdaki değerleri destekler:

- MQCNO_FASTPATH_BINDING
- MQCNO_STANDARD_BINDING
- MQCNO_SHARED_BINDING
- MQCNO_ISOLATED_BINDING
- MQCNO_SERIALIZE_CONN_TAG_QSG
- MQCNO_RESTRICT_CONN_TAG_QSG
- MQCNO_SERIALIZE_CONN_TAG_Q_MGR
- MQCNO_RESTRICT_CONN_TAG_Q_MGR

JMS için WebSphere MQ sınıfları tarafından kullanılan bağlantı seçeneklerini değiştirmek için, CONNOPTBağlantı Üreticisi özelliğini değiştirin.

Bağlantı seçeneklerine ilişkin ek bilgi için bkz. “MQCONNX çağrısını kullanarak kuyruk yöneticisine bağlanma” sayfa 200

Bağ tanımları aktarımcısını kullanmak için, kullanılmakta olan Java Runtime Environment 'ın, JMS için WebSphere MQ sınıflarının bağlanmakta olduğu kuyruk yöneticisinin Kodlanmış Karakter Takımı Tanıtıcısı 'nı (CCSID) desteklememesi gerekir.

Bir Java Runtime Environment tarafından desteklenen CCSID ' lerin nasıl belirleneceği ile ilgili ayrıntılar JMS için WebSphere MQ V7 sınıfları ya da JMS için WebSphere MQ V7 sınıfları kullanılırken, araştırıcı tanıtıcısı 21 olanWebSphere MQ FDC içinde bulunabilir.

Bağ tanımları, istemci kipi

Bu varsayılandır. Bu kipte bir kuyruk yöneticisine bağlanılırken, JMS uygulaması için bir WebSphere MQ sınıfları bağ tanımları kipinde bağlanmayı dener ve kuyruk yöneticisinin uygulamayla aynı makinede bulunmasını gerektirir. Bağlantı başarısız olursa, uygulama istemci kipinde bağlanmayı dener ve kuyruk yöneticisinin uygulamayla ya da uzaktan yerel olarak aynı makinede bulunmasına olanak tanır.

JMS uygulamalarına ilişkin WebSphere MQ sınıflarının istemci kipinde bağlanabilmesi için kuyruk yöneticinizin yapılandırılması

JMS uygulamaları için WebSphere MQ sınıflarının istemci kipinde bağlanabilmesi için kuyruk yöneticinizi yapılandırmak üzere, bir sunucu bağlantı kanalı tanımlaması yaratmanız ve bir dinleyici başlatmanız gerekir.

z/OSüzerinde, İstemci Eki özelliğinin kurulu olması gerekir.

Sunucu bağlantısı kanal tanımlaması yaratılması

Tüm altyapılarda, bir sunucu bağlantı kanalı tanımlaması yaratmak için MQSC komutu DEFINE CHANNEL ' ı kullanabilirsiniz. Aşağıdaki örneğe bakın:

```
DEFINE CHANNEL(JAVA.CHANNEL) CHLTYPE(SVRCONN) TRPTYPE(TCP)
```

IBM iüzerinde, aşağıdaki örnekte olduğu gibi, bunun yerine CRTMQMCHL CL komutunu kullanabilirsiniz:

```
CRTMQMCHL CHLNAME(JAVA.CHANNEL) CHLTYPE(*SVRCN)
                                TRPTYPE(*TCP)
                                MQMNAME(QMGRNAME)
```

Bu komutta *QMGRADı* , kuyruk yöneticinizin adıdır.

Linux ve Windowsüzerinde çalışan IBM WebSphere MQ Gezgini 'ni ya da z/OSüzerindeki işlemleri ve denetim panolarını kullanan bir sunucu bağlantı kanalı tanımlaması da yaratabilirsiniz.

Kanalın adı (JAVA.CHANNEL , uygulamanızın kuyruk yöneticisine bağlanmak için kullandığı bağlantı üreticisinin CHANNEL özelliği tarafından belirlenen kanal adıyla aynı olmalıdır. CHANNEL özelliğinin varsayılan değeri SYSTEM.DEF.SVRCONN.

Dinleyici başlatma

Önceden başlatılmamışsa, kuyruk yöneticiniz için bir dinleyici başlamanız gerekir.

Tüm altyapılarda, bir dinleyici başlatmak için MQSC komut START DINLEYICISINI kullanabilirsiniz; ancak, z/OSdışında, önce MQSC komutunu DEFINE LISTENER kullanarak bir dinleyici nesnesi yaratmanız gerekir. Aşağıdaki örneğe bakın:

```
DEFINE LISTENER(LISTENER.TCP) TRPTYPE(TCP) PORT(1414)
START LISTENER(LISTENER.TCP)
```

On UNIX, Linux, and Windows systems, you can also use the control command **Runmqlsr** to start a listener, as in the following example:

```
runmqlsr -t tcp -p 1414 -m QMgrName
```

Bu komutta, *QMgrName* kuyruk yöneticinizin adıdır.

You can also start a listener using WebSphere MQ Explorer, which runs on Linux and Windows, or the operations and control panels on z/OS.

İletişiminizin dinlemede olduğu kapının numarası, uygulamanızın kuyruk yöneticisine bağlanmak için kullandığı bağlantı üreticisinin PORT (Kapı) özelliği tarafından belirlenen kapı numarasıyla aynı olmalıdır. PORT (Kapı) özelliğinin varsayılan değeri 1414 'tür.

JMS için WebSphere MQ sınıfları için noktadan noktaya kuruluş doğrulama sınaması

JMS için WebSphere MQ sınıflarıyla birlikte bir noktadan noktaya kuruluş doğrulama sınaması (IVT) programı sağlanır. Program, bağ tanımlarında ya da istemci kipinde bir kuyruk yöneticisine bağlanır ve kuyruğa SYSTEM.DEFAULT.LOCAL.QUEUEve daha sonra, iletiyi kuyruktan alır. Program, yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratabilir ve yapılandırabilir ya da bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanabilir.

Sınama bağımsız olduğundan ve bir dizin hizmetinin kullanılmasını gerektirmediğinden, önce JNDI kullanmadan kuruluş doğrulama sınamasını çalıştırın. Denetlenen nesnelere ilişkin açıklamalar için bkz. “JMS nesne tipleri” sayfa 900.

JNDI kullanmadan noktadan noktaya kuruluş doğrulama sınaması

Bu testte, IVT programı yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratır ve yapılandırır ve JNDI kullanmaz.

IVT programını çalıştırmak için bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde IVTRun ve Windowsüzerinde IVTRun.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` altdizininde bulunur.

Sınamayı bağ tanımlama kipinde çalıştırmak için şu komutu girin:

```
IVTRun -nojndi [-m qmgr] [-v providerVersion] [-t]
```

Testi istemci kipinde çalıştırmak için önce kuyruk yöneticisini “Örnek programların hazırlanması ve çalıştırılması” sayfa 104içinde açıklandığı gibi ayarlayın. Kullanılacak kanalın varsayılan değeri **SYSTEM.DEF.SVRCONN** ve kullanılacak kuyruğun **SYSTEM.DEFAULT.LOCAL.QUEUE**olduğunu unutmayın ve aşağıdaki komutu girin:

```
IVTRun -nojndi -client -m qmgr -host hostname [-port port] [-channel channel]
       [-v providerVersion] [-ccsid ccsid] [-t]
```

z/OS sistemlerinde eşdeğer bir komut dosyası sağlanmaz, ancak aşağıdaki komutu kullanarak Java sınıfını doğrudan çağırarak IVT ' yi bağ tanımlama kipinde çalıştırabilirsiniz:

```
java com.ibm.mq.jms.MQJMSIVT -nojndi [-m qmgr] [-v providerVersion] [-t]
```

Sınıf yolu com.ibm.mqjms.jariçermelidir.

Komutlara ilişkin değiştirgeler aşağıdaki anlamlara sahiptir:

-m *qmgr*
IVT programının bağlandığı kuyruk yöneticisinin adı. Sınamayı bağ tanımlama kipinde çalıştırır ve bu değiştirgeyi atlarsanız, IVT programı varsayılan kuyruk yöneticisine bağlanır.

-host *anasistemadı*
Kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.

-port *kapı*
Kuyruk yöneticisinin dinleyicisinin dinlediği kapının numarası. Varsayılan değer 1414değeridir.

-channel *kanal*
IVT programının kuyruk yöneticisine bağlanmak için kullandığı MQI kanalının adı. Varsayılan değer `SYSTEM.DEF.SVRCONN`değeridir.

-v *providerVersion*
IVT programının bağlanmayı beklediği kuyruk yöneticisinin yayın düzeyi.

Bu değiştirge, bir MQQueueConnectionFactory nesnesinin PROVIDERVERSION özelliğini ayarlamak için kullanılır ve geçerli değerler PROVIDERVERSION özelliğininkiyle aynıdır. Bu nedenle, geçerli değerleri de içinde olmak üzere, bu parametreyle ilgili daha fazla bilgi için, IBM WebSphere MQ classes for JMS nesnelerinin özellikleri içindeki PROVIDERVERSION özelliğinin tanımına bakın.

Varsayılan değer `unspecified`değeridir.

-ccsid *ccsid*
Bağlantı tarafından kullanılacak kodlanmış karakter takımının ya da kod sayfasının tanıtıcısı (CCSID). Varsayılan değer `819`değeridir.

-t
İzleme açık. Varsayılan olarak izleme kapalıdır.

Başarılı bir sınama, aşağıdaki örnek çıkışa benzer bir çıkış üretir:

```
5724-H72, 5655-R36, 5724-L26, 5655-L82 (c) Copyright IBM Corp. 2008, 2024. All
Rights Reserved.
Websphere MQ classes for Java(tm) Message Service 7.0
Installation Verification Test

Creating a QueueConnectionFactory
Creating a Connection
Creating a Session
Creating a Queue
Creating a QueueSender
Creating a QueueReceiver
Creating a TextMessage
Sending the message to SYSTEM.DEFAULT.LOCAL.QUEUE
Reading the message back again

Got message
  JMSMessage class: jms_text
  JMSType:          null
  JMSDeliveryMode:  2
  JMSExpiration:    0
  JMSPriority:      4
  JMSMessageID:     ID:414d5120514d5f6d627720202020202001edb14620005e03
  JMSTimestamp:     1187170264000
  JMSCorrelationID: null
  JMSDestination:   queue:///SYSTEM.DEFAULT.LOCAL.QUEUE
  JMSReplyTo:       null
  JMSRedelivered:   false
    JMSXUserID: mwhite
    JMS_IBM_Encoding: 273
    JMS_IBM_PutApplType: 28
    JMSXAppID: WebSphere MQ Client for Java
    JMSXDeliveryCount: 1
    JMS_IBM_PutDate: 20070815
    JMS_IBM_PutTime: 09310400
    JMS_IBM_Format: MQSTR
    JMS_IBM_MsgType: 8
A simple text message from the MQJMSIVT
Reply string equals original string
Closing QueueReceiver
Closing QueueSender
Closing Session
Closing Connection
IVT completed OK
IVT finished
```

JNDI kullanan noktadan noktaya kuruluş doğrulama sınaması

Bu testte, IVT programı bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanır.

Sınamayı çalıştırmadan önce, LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) sunucusuna ya da yerel dosya sistemine dayalı bir dizin hizmeti yapılandırmanız gerekir. WebSphere MQ JMS yönetim aracını, yönetilen nesneleri depolamak için dizin hizmetini kullanabilecek şekilde yapılandırmanız da gerekir. Bu önkoşullar hakkında daha fazla bilgi için bkz. “JMS için WebSphere MQ sınıflarına ilişkin önkoşullar” sayfa 689. WebSphere MQ JMS yönetim aracının nasıl yapılandırılacağı hakkında bilgi için bkz. “JMS Denetimi aracının yapılandırılması” sayfa 897.

IVT programı, dizin hizmetinden bir MQQueueConnectionFactory nesnesi ve bir MQQueue nesnesini almak için JNDI ' yı kullanabilmelidir. Bu yönetilen nesneleri sizin için yaratmak üzere bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde IVTSetup ve Windowsüzerinde IVTSetup.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` altdizininde bulunur. Komut dosyasını çalıştırmak için şu komutu girin:

```
IVTSetup
```

Komut dosyası, yönetilen nesneleri yaratmak için WebSphere MQ JMS yönetim aracını çağırır.

MQQueueConnectionFactory nesnesi için ivtQCF adı verilir ve tüm özellikleri için varsayılan değerlerle yaratılır; bu, IVT programının bağ tanımlama kipinde çalıştığı ve varsayılan kuyruk yöneticisine bağlandığı anlamına gelir. IVT programının istemci kipinde çalışmasını ya da varsayılan kuyruk yöneticisinden başka bir kuyruk yöneticisine bağlanmasını istiyorsanız, MQQueueConnectionFactory nesnesinin uygun özelliklerini değiştirmek için WebSphere MQ JMS yönetim aracını ya da WebSphere MQ Explorer 'ı kullanmanız gerekir. WebSphere MQ JMS yönetim aracının nasıl kullanılacağına ilişkin bilgi için bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896. WebSphere MQ Explorer 'ın nasıl kullanılacağına ilişkin bilgi için WebSphere MQ Explorer ile sağlanan yardıma bakın.

MQQueue nesnesi için ivtQ adıyla bağ tanımlandı ve SYSTEM.DEFAULT.LOCAL.QUEUE.

Denetlenen nesneleri yarattığınızda, IVT programını çalıştırabilirsiniz. Sınamayı JNDI kullanarak çalıştırmak için şu komutu girin:

```
IVTRun -url "providerURL" [-icf initCtxFact] [-t]
```

Komuttaki değiştirgeler aşağıdaki anlamlara sahiptir:

-url "*providerURL*"
Dizin hizmetinin URL 'si. URL adresi aşağıdaki biçimlerden birine sahip olabilir:

- `ldap://`*`hostname`*`/`*`contextName`*, LDAP sunucusuna dayalı bir dizin hizmeti için
- `file:/`*`directoryPath`*, yerel dosya sistemine dayalı bir dizin hizmeti için

URL ' yi tırnak işareti (") içine almanız gerektiğini unutmayın.

-icf *initCtxOlgu*
Aşağıdaki değerlerden biri olması gereken ilk bağlam üreticisinin sınıf adı:

- `com.sun.jndi.ldap.LdapCtxFactory`, LDAP sunucusuna dayalı bir dizin hizmeti için. Bu varsayılan değerdir.
- `com.sun.jndi.fscontext.RefFSContextFactory`, yerel dosya sistemine dayalı bir dizin hizmeti için.

-t
İzleme açık. Varsayılan olarak izleme kapalıdır.

Başarılı bir test, JNDI kullanmadan başarılı bir test için buna benzer bir çıkış üretir. Ana fark, çıkışın bir MQQueueConnectionFactory nesnesini ve bir MQQueue nesnesini almak için JNDI kullandığını göstermesi.

Kesinlikle gerekli olmasa da, IVTSetup komut dosyası tarafından yaratılan yönetilen nesneleri silerek testten sonra toparlamak iyi bir uygulamadır. Bu amaçla bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde IVTTidy ve Windowsüzerinde IVTTidy.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` alt dizininde bulunur.

Noktadan noktaya kuruluş doğrulama sınaması için sorun belirleme

Kuruluş doğrulama sınaması aşağıdaki nedenlerden ötürü başarısız olabilir:

- IVT programı bir sınıf bulamadığını belirten bir ileti yazarsa, sınıf yolunuzun “JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri” sayfa 694içinde açıklandığı gibi doğru ayarlandığını doğrulayın.
- Sınama şu iletiyle başarısız olabilir:

  ```
  Failed to connect to queue manager 'qmgr' with connection mode 'connMode'
  and host name 'hostname'
  ```

  ve ilişkili neden kodu 2059. İletideki değişkenler aşağıdaki anlamlara sahiptir:

  qmgr
  IVT programının bağlanmaya çalıştığı kuyruk yöneticisinin adı. IVT programı bağ tanımlama kipinde varsayılan kuyruk yöneticisine bağlanmaya çalışıyorsa, araya bu ileti boş olur.

  connMode
  `Bindings` ya da `Client`olan bağlantı kipi.

  hostname
  Kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.

  Bu ileti, IVT programının bağlanmaya çalıştığı kuyruk yöneticisinin kullanılamadığı anlamına gelir. Kuyruk yöneticisinin çalışıp çalışmadığını denetleyin ve IVT programı varsayılan kuyruk yöneticisine bağlanmaya çalışıyorsa, kuyruk yöneticisinin sisteminiz için varsayılan kuyruk yöneticisi olarak tanımlandığını doğrulayın.
- Sınama şu iletiyle başarısız olabilir:

  ```
  Failed to open MQ queue 'SYSTEM.DEFAULT.LOCAL.QUEUE'
  ```

  Bu ileti, kuyruğun SYSTEM.DEFAULT.LOCAL.QUEUE , IVT programının bağlı olduğu kuyruk yöneticisinde yok. Diğer bir seçenek olarak, kuyruk varsa, IVT programı ileti koymak ve almak için etkinleştirilmediği için kuyruğu açamaz. Kuyruğun var olup olmadığını ve ileti yerleştirmek ve almak için etkinleştirilip etkinleştirilmediğini denetleyin.
- Sınama şu iletiyle başarısız olabilir:

  ```
  Unable to bind to object
  ```

  Bu ileti, LDAP sunucusuna yönelik bir bağlantı olduğu, ancak LDAP sunucusunun doğru yapılandırılmadığı anlamına gelir. LDAP sunucusu Java nesnelerini saklamak için yapılandırılmamış ya da nesnelere ya da soneke ilişkin izinler doğru değil. Bu durumda daha fazla yardım için LDAP sunucunuza ilişkin belgelere bakın.
- Sınama şu iletiyle başarısız olabilir:

  ```
  The security authentication was not valid that was supplied for
  QueueManager 'qmgr' with connection mode 'Client' and host name 'hostname'
  ```

  Bu ileti, kuyruk yöneticisinin sisteminizden gelen bir istemci bağlantısını kabul edecek şekilde doğru ayarlanmadığı anlamına gelir. Ayrıntılar için bkz. “Örnek programların hazırlanması ve çalıştırılması” sayfa 104.

JMS için WebSphere MQ sınıfları için yayınlama/abone olma kuruluş doğrulama sınaması

JMS için WebSphere MQ sınıflarıyla birlikte bir yayınlama/abone olma kuruluş doğrulama testi (IVT) programı sağlanır. Program, bağ tanımları ya da istemci kipinde bir kuyruk yöneticisine bağlanır, bir konuya abone olur, konuyla ilgili bir ileti yayınlar ve daha sonra, yeni yayınladığı iletiyi alır. Program, yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratabilir ve yapılandırabilir ya da bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanabilir.

Sınama bağımsız olduğundan ve bir dizin hizmetinin kullanılmasını gerektirmediğinden, önce JNDI kullanmadan kuruluş doğrulama sınamasını çalıştırın. Denetlenen nesnelere ilişkin açıklamalar için bkz. "JMS nesne tipleri" sayfa 900.

JNDI kullanmadan yayınlama/abone olma kuruluş doğrulama sınaması

Bu testte, IVT programı yürütme sırasında dinamik olarak gerektirdiği tüm nesneleri yaratır ve yapılandırır ve JNDI kullanmaz.

IVT programını çalıştırmak için bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde PSIVTRun ve Windowsüzerinde PSIVTRun.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` altdizininde bulunur.

Sınamayı bağ tanımlama kipinde çalıştırmak için şu komutu girin:

```
PSIVTRun -nojndi [-m qmgr] [-bqm brokerQmgr] [-v providerVersion] [-t]
```

Sınamayı istemci kipinde çalıştırmak için, önce kuyruk yöneticisini "Örnek programların hazırlanması ve çalıştırılması" sayfa 104 içinde açıklandığı gibi ayarlayın ve kullanılacak kanalın varsayılan olarak SYSTEM.DEF.SVRCONN, ardından şu komutu girin:

```
PSIVTRun -nojndi -client -m qmgr -host hostname [-port port] [-channel channel]
         [-bqm brokerQmgr] [-v providerVersion] [-ccsid ccsid] [-t]
```

Komutlara ilişkin değiştirgeler aşağıdaki anlamlara sahiptir:

-m *qmgr*
IVT programının bağlandığı kuyruk yöneticisinin adı. Sınamayı bağ tanımlama kipinde çalıştırır ve bu değiştirgeyi atlarsanız, IVT programı varsayılan kuyruk yöneticisine bağlanır.

-host *anasistemadı*
Kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.

-port *kapı*
Kuyruk yöneticisinin dinleyicisinin dinlediği kapının numarası. Varsayılan değer 1414değeridir.

-channel *kanal*
IVT programının kuyruk yöneticisine bağlanmak için kullandığı MQI kanalının adı. Varsayılan değer `SYSTEM.DEF.SVRCONN`değeridir.

-bqm *brokerQmgr*
Aracının çalıştığı kuyruk yöneticisinin adı. Varsayılan değer, IVT programının bağlandığı kuyruk yöneticisinin adıdır.

Bu değiştirge yalnızca, -v değiştirgesi 7 'den küçük bir kuyruk yöneticisi sürüm numarası belirtiyorsa ve yayınlama/abone olma aracısı olarak WebSphere Event Broker ya da WebSphere Message Broker kullanıyorsanız anlamlıdır.

-v *providerVersion*
IVT programının bağlanmayı beklediği kuyruk yöneticisinin yayın düzeyi.

Bu değiştirge, bir MQTopicConnectionFactory nesnesinin PROVIDERVERSION özelliğini ayarlamak için kullanılır ve geçerli değerler PROVIDERVERSION özelliğininkiyle aynıdır. Bu nedenle, geçerli değerleri de içinde olmak üzere, bu parametreyle ilgili daha fazla bilgi için, IBM WebSphere MQ classes for JMS nesnelerinin özellikleri içindeki PROVIDERVERSION özelliğinin tanımına bakın.

Varsayılan değer `unspecified`değeridir.

-ccsid *ccsid*
Bağlantı tarafından kullanılacak kodlanmış karakter takımının ya da kod sayfasının tanıtıcısı (CCSID). Varsayılan değer 819değeridir.

-t
İzleme açık. Varsayılan olarak izleme kapalıdır.

Başarılı bir sınama, aşağıdaki örnek çıkışa benzer bir çıkış üretir:

```
5724-H72, 5655-R36, 5724-L26, 5655-L82 (c) Copyright IBM Corp. 2008, 2024. All
Rights Reserved.
Websphere MQ classes for Java(tm) Message Service 7.0
Publish/Subscribe Installation Verification Test

Creating a TopicConnectionFactory
Creating a Connection
Creating a Session
Creating a Topic
Creating a TopicPublisher
Creating a TopicSubscriber
Creating a TextMessage
Adding text
Publishing the message to topic://MQJMS/PSIVT/Information
Waiting for a message to arrive [5 secs max]...

Got message:
  JMSMessage class: jms_text
  JMSType:          null
  JMSDeliveryMode:  2
  JMSExpiration:    0
  JMSPriority:      4
  JMSMessageID:     ID:414d5120514d5f6d627720202020202001edb14620006706
  JMSTimestamp:     1187182520203
  JMSCorrelationID: ID:414d5120514d5f6d627720202020202001edb14620006704
  JMSDestination:   topic://MQJMS/PSIVT/Information
  JMSReplyTo:       null
  JMSRedelivered:   false
    JMSXUserID: mwhite
    JMS_IBM_Encoding: 273
    JMS_IBM_PutApplType: 26
    JMSXAppID: QM_mbw
    JMSXDeliveryCount: 1
    JMS_IBM_PutDate: 20070815
    JMS_IBM_ConnectionID: 414D5143514D5F6D627720202020202001EDB14620006601
    JMS_IBM_PutTime: 12552020
    JMS_IBM_Format: MQSTR
    JMS_IBM_MsgType: 8
A simple text message from the MQJMSPSIVT program
Reply string equals original string
Closing TopicSubscriber
Closing TopicPublisher
Closing Session
Closing Connection
PSIVT finished
```

JNDI kullanarak yayınlama/abone olma kuruluş doğrulama sınaması

Bu testte, IVT programı bir dizin hizmetinden yönetilen nesneleri almak için JNDI ' yı kullanır.

Sınamayı çalıştırmadan önce, LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) sunucusuna ya da yerel dosya sistemine dayalı bir dizin hizmeti yapılandırmanız gerekir. WebSphere MQ JMS yönetim aracını, yönetilen nesneleri depolamak için dizin hizmetini kullanabilecek şekilde yapılandırmanız da gerekir. Bu önkoşullar hakkında daha fazla bilgi için bkz. “JMS için WebSphere MQ sınıflarına ilişkin önkoşullar” sayfa 689. WebSphere MQ JMS yönetim aracının nasıl yapılandırılacağı hakkında bilgi için bkz. “JMS Denetimi aracının yapılandırılması” sayfa 897.

IVT programının, dizin hizmetinden bir MQTopicConnectionFactory nesnesi ve bir MQTopic nesnesini almak için JNDI ' yı kullanabilmesi gerekir. Bu yönetilen nesneleri sizin için yaratmak üzere bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde IVTSetup ve Windowsüzerinde IVTSetup.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` altdizininde bulunur. Komut dosyasını çalıştırmak için şu komutu girin:

```
IVTSetup
```

Komut dosyası, yönetilen nesneleri yaratmak için WebSphere MQ JMS yönetim aracını çağırır.

MQTopicConnectionFactory nesnesi için ivtTCF adı verilir ve tüm özellikleri için varsayılan değerlerle yaratılır; bu, IVT programının bağ tanımlama kipinde çalıştığı, varsayılan kuyruk yöneticisine bağlandığı ve yerleşik yayınlama/abone olma işlevini kullandığı anlamına gelir. IVT programının istemci kipinde çalışmasını istiyorsanız, varsayılan kuyruk yöneticisinden başka bir kuyruk yöneticisine bağlanın ya da yerleşik yayınlama/abone olma işlevi yerine WebSphere Event Broker ya da WebSphere Message Broker olanağını kullanın. MQTopicConnectionFactory nesnesinin uygun özelliklerini değiştirmek için WebSphere MQ JMS yönetim aracını ya da WebSphere MQ Explorer aracını kullanmalısınız. WebSphere MQ JMS yönetim aracının nasıl kullanılacağına ilişkin bilgi için bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896. WebSphere MQ Explorer 'ın nasıl kullanılacağına ilişkin bilgi için WebSphere MQ Explorer ile sağlanan yardıma bakın.

MQTopic nesnesi için ivtT adı bağlanır ve MQJMS/PSIVT/Information değerine sahip TOPIC özelliği dışında, tüm özellikleri için varsayılan değerlerle yaratılır.

Denetlenen nesneleri yarattığınızda, IVT programını çalıştırabilirsiniz. Sınamayı JNDI kullanarak çalıştırmak için şu komutu girin:

```
PSIVTRun -url "providerURL" [-icf initCtxFact] [-t]
```

Komuttaki değiştirgeler aşağıdaki anlamlara sahiptir:

-url "*providerURL*"
Dizin hizmetinin URL 'si. URL adresi aşağıdaki biçimlerden birine sahip olabilir:

- `ldap://hostname/contextName`, LDAP sunucusuna dayalı bir dizin hizmeti için
- `file:/directoryPath`, yerel dosya sistemine dayalı bir dizin hizmeti için

URL ' yi tırnak işareti (") içine almanız gerektiğini unutmayın.

-icf *initCtxOlgu*
Aşağıdaki değerlerden biri olması gereken ilk bağlam üreticisinin sınıf adı:

- `com.sun.jndi.ldap.LdapCtxFactory`, LDAP sunucusuna dayalı bir dizin hizmeti için. Bu varsayılan değerdir.
- `com.sun.jndi.fscontext.RefFSContextFactory`, yerel dosya sistemine dayalı bir dizin hizmeti için.

-t
İzleme açık. Varsayılan olarak izleme kapalıdır.

Başarılı bir test, JNDI kullanmadan başarılı bir test için buna benzer bir çıkış üretir. Ana fark, çıkışın bir MQTopicConnectionFactory nesnesini ve bir MQTopic nesnesini almak için JNDI kullandığını göstermesi.

Kesinlikle gerekli olmasa da, IVTSetup komut dosyası tarafından yaratılan yönetilen nesneleri silerek testten sonra toparlamak iyi bir uygulamadır. Bu amaçla bir komut dosyası sağlanır. Komut dosyası, UNIX and Linux sistemlerinde IVTTidy ve Windowsüzerinde IVTTidy.bat olarak adlandırılır ve JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` alt dizininde bulunur.

Yayınlama/abone olma kuruluş doğrulama testi için sorun belirleme

Kuruluş doğrulama sınaması aşağıdaki nedenlerden ötürü başarısız olabilir:

- IVT programı bir sınıf bulamadığını belirten bir ileti yazarsa, sınıf yolunuzun “JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri” sayfa 694içinde açıklandığı gibi doğru ayarlandığını doğrulayın.
- Sınama şu iletiyle başarısız olabilir:

  ```
  Failed to connect to queue manager 'qmgr' with
  connection mode 'connMode' and host name 'hostname'
  ```

  ve ilişkili neden kodu 2059. İletideki değişkenler aşağıdaki anlamlara sahiptir:

qmgr
IVT programının bağlanmaya çalıştığı kuyruk yöneticisinin adı. IVT programı bağ tanımlama kipinde varsayılan kuyruk yöneticisine bağlanmaya çalışıyorsa, araya bu ileti boş olur.

connMode
`Bindings` ya da `Client`olan bağlantı kipi.

hostname
Kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.

Bu ileti, IVT programının bağlanmaya çalıştığı kuyruk yöneticisinin kullanılamadığı anlamına gelir. Kuyruk yöneticisinin çalışıp çalışmadığını denetleyin ve IVT programı varsayılan kuyruk yöneticisine bağlanmaya çalışıyorsa, kuyruk yöneticisinin sisteminiz için varsayılan kuyruk yöneticisi olarak tanımlandığını doğrulayın.

- Sınama şu iletiyle başarısız olabilir:

```
Unable to bind to object
```

Bu ileti, LDAP sunucusuna yönelik bir bağlantı olduğu, ancak LDAP sunucusunun doğru yapılandırılmadığı anlamına gelir. LDAP sunucusu Java nesnelerini saklamak için yapılandırılmamış ya da nesnelere ya da soneke ilişkin izinler doğru değil. Bu durumda daha fazla yardım için LDAP sunucunuza ilişkin belgelere bakın.

- Sınama şu iletiyle başarısız olabilir:

```
The security authentication was not valid that was supplied for
QueueManager 'qmgr' with connection mode 'Client' and host name 'hostname'
```

Bu ileti, kuyruk yöneticisinin sisteminizden gelen bir istemci bağlantısını kabul edecek şekilde doğru ayarlanmadığı anlamına gelir. Daha fazla bilgi için bkz “Örnek programların hazırlanması ve çalıştırılması” sayfa 104.

WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı

IVT programı bir EAR dosyası olarak sağlanır. Programı kullanmak için, programı konuşlandırmanız ve JCA kaynakları olarak bazı nesneler tanımlamanız gerekir.

Kuruluş doğrulama sınaması (IVT) programı, wmq.jmsra.ivt.earadlı kurumsal arşiv (EAR) dosyası olarak sağlanır. This file is installed with WebSphere MQ classes for JMS in the same directory as the WebSphere MQ resource adapter RAR file, wmq.jmsra.rar. Bu dosyaların nereye takıldığı hakkında bilgi için bkz. “ WebSphere MQ Resource Adapter ürününün kuruluşu” sayfa 705.

Uygulama sunucunuzda IVT programını konuşlandırmanız gerekir. IVT programı, bir sunucu uygulamacığı ve bir WebSphere MQ kuyruğundan ileti gönderilebileceğini ve bu kuyruktan alınabileceğini test eden bir MDB içerir. Optionally, you can use the IVT program to verify that the WebSphere MQ resource adapter has been correctly configured to support distributed transactions.

IVT programını çalıştırabilmeniz için önce bir ConnectionFactory nesnesi, bir Kuyruk nesnesi ve JCA kaynakları olarak büyük olasılıkla bir Etkinleştirme Belirtimi nesnesi tanımlamanız ve uygulama sunucunuzun bu tanımlamalardan JMS nesneleri yaratmasını ve bunları bir JNDI ad alanına bağlamalarını sağlamanız gerekir. Nesnelerin özelliklerini seçebilirsiniz, ancak aşağıdaki özellik kümesi basit bir örnektir:

ConnectionFactory nesnesi

```
channel:               SYSTEM.DEF.SVRCONN
hostName:              localhost
port:                  1414
queueManager:          ExampleQM
transportType:         CLIENT
```

Kuyruk nesnesi

```
baseQueueManagerName:  ExampleQM
baseQueueName:         TEST.QUEUE
```

Varsayılan olarak IVT programı, jms/ivt/IVTCF adı ve jms/ivt/IVTQueue adı ile bağ tanımlanacak bir Kuyruk nesnesi olan JNDI ad alanında bir ConnectionFactory nesnesinin bağlanabilmesini bekler. Farklı adlar kullanabilirsiniz, ancak bunu yapmak için, IVT programının başlangıç sayfasındaki nesnelerin adlarını girmeniz ve EAR dosyasını uygun bir şekilde değiştirmelisiniz.

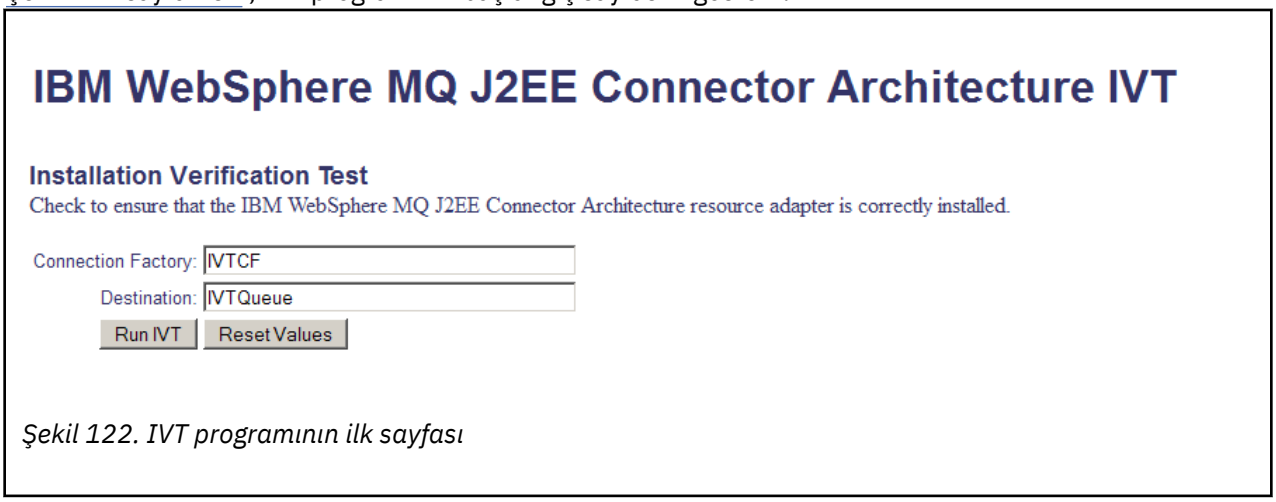
IVT programını konuşlandırdıktan ve uygulama sunucusu JMS nesnelerini yarattıktan ve bunları JNDI ad alanına bağladıktan sonra, Web tarayıcınıza aşağıdaki biçimde bir URL girerek IVT programını başlatabilirsiniz:

```
http://app_server_host:port/WMQ_IVT/
```

Burada *app_server_host* , uygulama sunucunuzun çalıştırıldığı sistemin IP adresi ya da anasistem adı ve *kapı* , uygulama sunucusunun dinlediği TCP kapısının numarasıdır. Örnek:

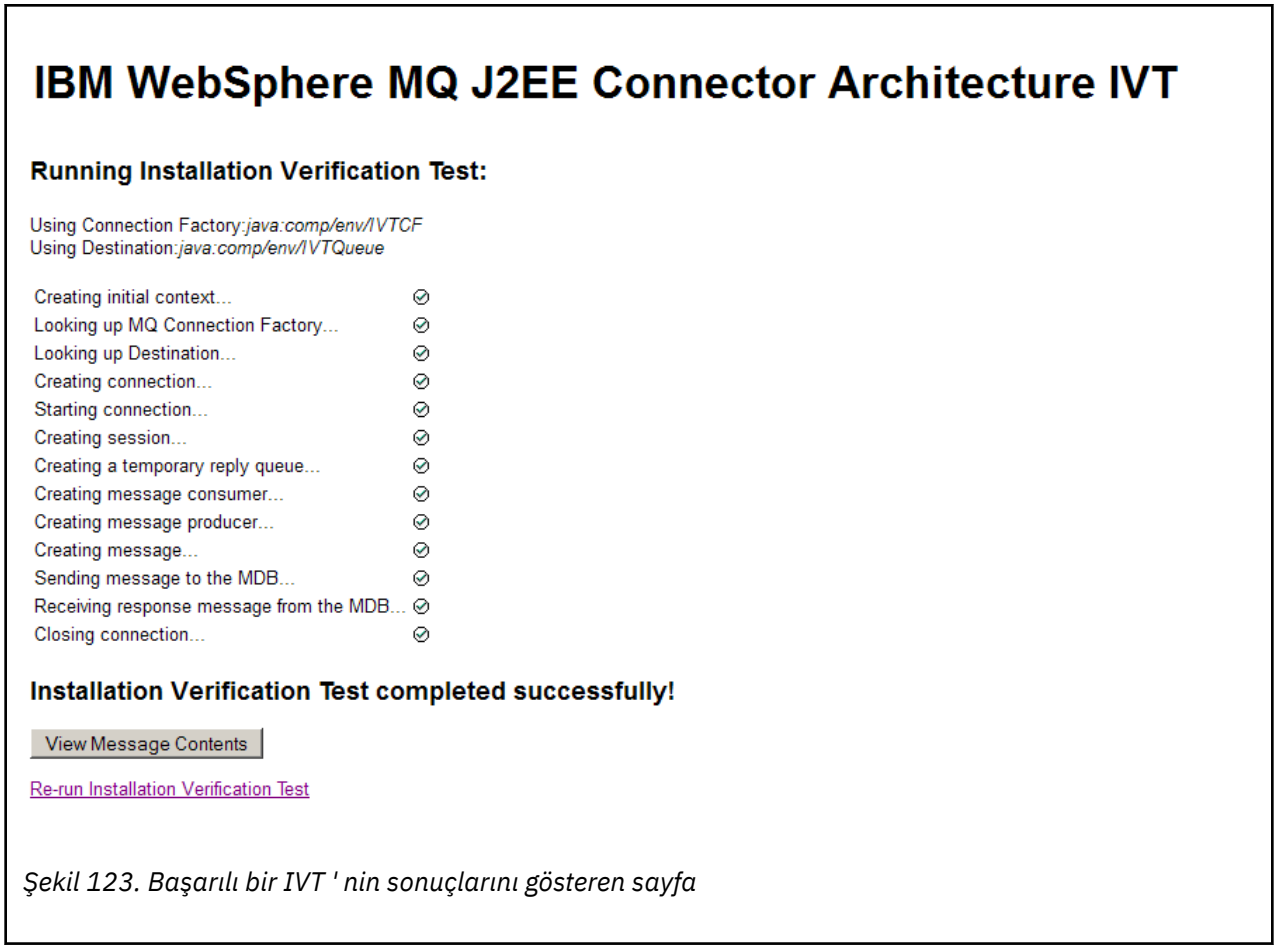
```
http://localhost:9080/WMQ_IVT/
```

Şekil 122 sayfa 751 , IVT programının başlangıç sayfasını gösterir.



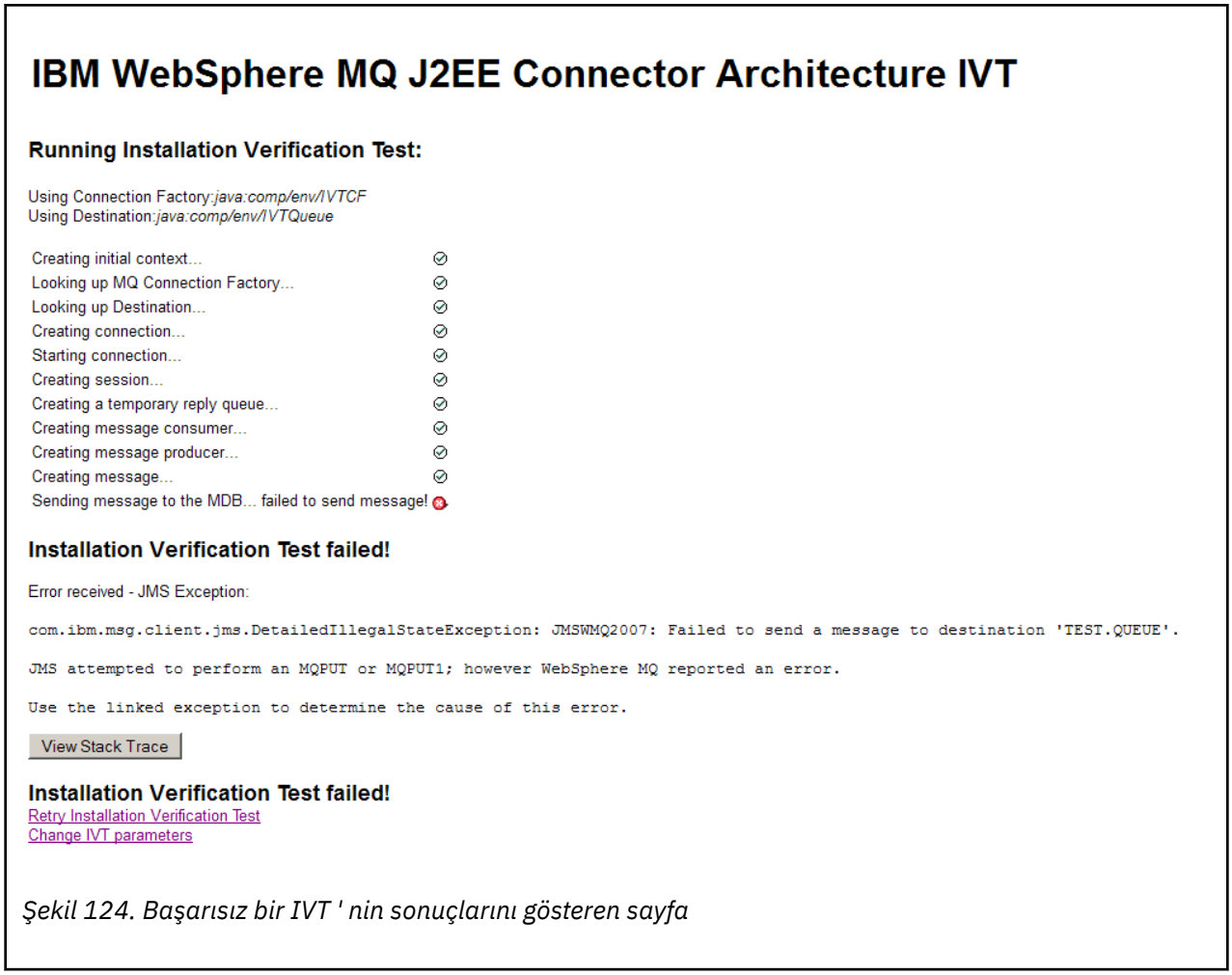
Şekil 122. IVT programının ilk sayfası

Sınamayı çalıştırmak için **IVT 'yi çalıştır**' ı tıklatın. Şekil 123 sayfa 752 IVT başarılı olursa görüntülenen sayfayı gösterir.



Şekil 123. Başarılı bir IVT ' nin sonuçlarını gösteren sayfa

IVT başarısız olursa, Şekil 124 sayfa 753 içinde gösterilen bir sayfa görüntülenir. Hatanın nedenine ilişkin daha fazla bilgi edinmek için **Yığın İzlemesini Görüntüle**düğmesini tıklatın.



Şekil 124. Başarısız bir IVT ' nin sonuçlarını gösteren sayfa

JBoss ve WAS CE uygulama sunucularındaki IVT uygulamasını konuşlandırmak için sağlanan yardımcı program komut dosyalarına ilişkin ayrıntılı yönergeler ve bilgiler için bkz:

İlgili görevler

“WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması” sayfa 753
IBM WebSphere MQ kaynak bağdaştırıcısının kurulması ve WebSphere Application Server CE ' de kuruluş doğrulama sınaması (IVT) uygulaması çalıştırılması.

“Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756
IBM WebSphere MQ kaynak bağdaştırıcısını JBoss AS 5.1 ya da 6 olarak kurduktan sonra, kuruluş doğrulama sınaması (IVT) uygulamasını kurarak ve çalıştırarak kaynak bağdaştırıcısının kuruluşunu sınayarak sınayarak sınabilirsiniz.

WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması

IBM WebSphere MQ kaynak bağdaştırıcısının kurulması ve WebSphere Application Server CE ' de kuruluş doğrulama sınaması (IVT) uygulaması çalıştırılması.

Başlamadan önce

Bu görev, çalışmakta olan bir WebSphere Application Server CE sunucusuna sahip olduğunuzu ve bunun için standart yönetim görevlerini aşina olduğunuzu varsayar. Bu görev ayrıca, yerel sisteminizde bir IBM WebSphere MQ kuruluşu olduğunu ve standart denetim görevlerine alışkın olduğunuzu varsayar.

Bir IBM WebSphere MQ istemcisine bağlanmak için kaynak bağdaştırıcısı kullanıyorsanız ve dağıtılmış XA hareketleri gerekiyorsa, **Yalnızca istemci XA**olarak işaretlenmiş ek adımları izlemeniz gerekir.

1. Create a queue manager called ExampleQM, and set it up as described in “Örnek programların hazırlanması ve çalıştırılması” sayfa 104 noting that the listener should be started on port 1414, the channel to be used is called SYSTEM.DEF.SVRCONN and the queue used by the IVT application is named TEST.QUEUE. Ayrıca, bu uygulamanın geçici bir yanıt kuyruğu yaratabilmesi için, SYSTEM.DEFAULT.MODEL.QUEUE model kuyruğunda DSP ve PUT yetkisi de verilmelidir. Farklı bir kuyruk yöneticisi, farklı bağlantı ayrıntıları ya da farklı bir kuyruk kullanmak istiyorsanız bkz. “Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması” sayfa 755.
2. Kaynak bağdaştırıcısı dosyasını (wmq.jmsra.rar), IVT uygulamasını (wmq.jmsra.ivt.ear) ve WAS_CE_jmsra_deployment_plan.xml ve WAS_CE_jmsra_ivt_deployment_plan.xml deployment plan files' yi edinin. Bu dosyaların yeriyle ilgili ayrıntılar için bkz. “ WebSphere MQ Resource Adapter ürününün kuruluşu” sayfa 705.

Bağ tanımlarına ve istemci kipi bağlantılarına ilişkin açıklamalar için bkz. “JMS için WebSphere MQ sınıflarına ilişkin bağlantı kipleri” sayfa 741.

İstemci kipi yerine farklı bir kuyruk, kuyruk yöneticisi, kapı, anasistem, kanal ya da bağ tanımları kipini kullanmak istiyorsanız, “Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması” sayfa 755başlıklı konuya bakın.

Yordam

1. **Yalnızca istemci XA**: WAS_CE_jmsra_deployment_plan.xml dosyasını kopyalarınızı düzenleyin.
   a) jms/ivt/IVTCF bağlantı tanımını bulun ve bağlantı üreticisi XA hareketi etkin durumda olacak şekilde değiştirin.
      i) NonXA bölümünü açıklama satırı yapın:

```
<conn:xa-transaction>
```

      ii) XA yapılanışı kısmını açıklama satırı yapın:

```
<conn:xa-transaction>
    <conn:transaction-caching/>
</conn:xa-transaction>
```

   b) Değişikliklerinizi kaydedin.
2. İsteğe bağlı: **Yalnızca istemci XA**: İşlem gerektirmesi için MDB ' nin derleme tanımlayıcısını değiştirin. Bu işlem, IVT uygulaması bu değişiklik yapılmadan çalışmaya devam etse de, IVT 'deki MDB' yi XA hareketine katılmak üzere zorlar.
   a) wmq.jmsra.ivt.ear dosyasını açın.
   b) İçinde WMQ_IVT_MDB.jar dosyasını açın.
   c) META-INF/ejb-jar.xmldosyasını düzenleyin.
      i) Çevirme tanımlayıcısı içindeki satırı açıklama satırı yap ya da sil:

```
<trans-attribute>NotSupported</trans-attribute>
```

      ii) Yapıbirimi tanımlayıcısı içindeki satırı açıklama satırı olarak kaldırın:

```
<trans-attribute>Required</trans-attribute>
```

      iii) Değişikliklerinizi saklayın ve WMQ_IVT_MDB.jar dosyası içinde dosyayı güncelleyin.
      iv) Update the wmq.jmsra.ivt.ear file with the modified WMQ_IVT_MDB.jar file.
3. Değiştirilen devreye alma planı dosyasını kullanarak kaynak bağdaştırıcısını sunucunuza dağıt.
   a) Bu işlemi komut satırında yapmak için aşağıdaki WAS CE komutunu yazın:

```
deploy -u system -p manager deploy wmq.jmsra.rar WAS_CE_jmsra_deployment_plan.xml
```

b) Web denetimi arabirimini kullanarak **Uygulamalar > Deployer:**(Uygulamalar > Devreye Alma) öğelerini seçin.

i) Arşivi, wmq.jmsra.rar dosyası olacak şekilde ayarlayın.

ii) Planı, WAS_CE_jmsra_deployment_plan.xml dosyası olacak şekilde ayarlayın.

iii) 'kuruluştan sonra uygulamayı başlat' seçeneğinin belirlendiğinden emin olun.

iv) **Kur**'u tıklatın.

4. Sağlanan devreye alma planını kullanarak IVT uygulamasını sunucunuza dağıt.

a) Komut satırında, bu işlem aşağıdaki WAS CE komutu kullanılarak yapılabilir:

```
deploy -u system -p manager deploy wmq.jmsra.ivt.ear WAS_CE_jmsra_ivt_deployment_plan.xml
```

b) Web denetimi arabirimini kullanarak **Uygulamalar > Deployer:**(Uygulamalar > Devreye Alma) öğelerini seçin.

i) **Arşiv** ' i `wmq.jmsra.ivt.ear` dosyası olacak şekilde ayarlayın.

ii) **Plan** ' ı `WAS_CE_jmsra_ivt_deployment_plan.xml` dosyası olarak ayarlayın.

iii) **Uygulamayı kurduktan sonra başlat** seçeneğinin belirlendiğinden emin olun.

iv) **Kur**'u tıklatın.

5. IVT uygulamasını çalıştırın. Daha fazla ayrıntı için bkz. “ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750. WAS CE için varsayılan URL `http://localhost:8080/WMQ_IVT/` ' dir.

Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması

If you want to use a different queue, queue manager, port, host, channel, or use bindings mode instead of client mode, you must modify the IVT application and associated scripts in WebSphere Application Server CE before deploying the resource adapter or IVT application.

Bu görev hakkında

“WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması” sayfa 753' ta belirtilen bir yapılandırmadan farklı bir yapılandırmaya konuşlandırmak istiyorsanız, bu durumda, istemci kipi yerine farklı bir kuyruk, kuyruk yöneticisi, kapı, anasistem, kanal ya da bağ tanımları kipini kullanmak istiyorsanız, kaynak bağdaştırıcısı ya da IVT uygulamasını konuşlandırmadan önce aşağıdaki adımları gerçekleştirin.

Yordam

1. IVT uygulaması için kullanmak üzere farklı bir kuyruk yöneticisi ve kuyruk belirlemek istiyorsanız, kuyruk yöneticisi için değerleri ayarlayın ve `WAS_CE_jmsra_deployment_plan.xml`'ta kuyruğa girin; ayrıntılar için “Kuyruk yöneticisi ve kuyruk için değer ayarlanıyor” sayfa 756' a bakın.
2. İletiyle yönlendirilen Bean (MDB) için yapılandırmada farklı bir kuyruk yöneticisi ve kuyruk belirtmek istiyorsanız, ayrıntılar için `WAS_CE_jmsra_ivt_deployment_plan.xml`içinde kullandığınız kuyruk yöneticisine ve kuyruğa ilişkin değerleri ayarlayın; ayrıntılar için “MDB yapılandırması için değerler ayarlanıyor” sayfa 756' a bakın.
3. Kaynak bağdaştırıcısını, bağ tanımları kipinde IBM WebSphere MQ ' e bağlanmak için yapılandırıyorsanız, JNI kitaplıklarının sistem yolunda ya da WAS CE için yol üzerinde olduğundan emin olun. Daha fazla bilgi için, bkz. “WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması” sayfa 753.
4. Kaynak bağdaştırıcısını önceden konuşlandırdıysanız, aşağıdaki komutu kullanarak ayarları değiştirmek için, bunu değiştirilen konuşlandırma planıyla yeniden devreye alabilirsiniz:

```
deploy --user system --password manager redeploy wmq.jmsra.rar
WAS_CE_jmsra_deployment_plan.xml
```

Sonraki adım

Continue deploying the resource adapter as described in “WAS CE ' de MQ kaynak bağdaştırıcısının kurulması ve sınanması” sayfa 753.

Kuyruk yöneticisi ve kuyruk için değer ayarlanıyor

`WAS_CE_jmsra_deployment_plan.xml`' ta kullanmakta olduğunuz kuyruk yöneticisi ve kuyruk için değerlerin nasıl ayarlanacak olduğunu açıklar.

Yordam

`WAS_CE_jmsra_deployment_plan.xml`' ta, IVT uygulaması için kullandığınız kuyruk yöneticisine ve kuyruğa ilişkin değerleri ayarlayın.

Jms/ivt/IVTCF bağlantı tanımı için:

1. queueManager öğesinin değerini, kuyruk yöneticinizin adı olacak şekilde ayarlayın.
2. Bir istemci bağlantısı kullanıyorsanız, çeşitli istemci bağlantı öğelerinin değerini, kuyruk yöneticinize yönelik bir bağlantı için uygun olacak şekilde ayarlayın.
3. Bağ tanımları bağlantısı kullanıyorsanız:
   a. transportType öğesinin değerini bağ tanımlarını (BATTACT) olarak ayarlayın.
   b. Çeşitli istemci bağlantısı öğelerini dışarı çıkarın ya da silin.
4. Jms/iv/IVTQueue ileti hedefi için, baseQueueName öğesinin değerini, IVT uygulaması için yarattığınız kuyruğun adı olacak şekilde ayarlayın.
5. Değişikliklerinizi kaydedin.

MDB yapılandırması için değerler ayarlanıyor

`WAS_CE_jmsra_deployment_plan.xml`' ta MDB yapılandırması için değerlerin nasıl ayarlanacak olduğunu açıklar.

Yordam

`WAS_CE_jmsra_ivt_deployment_plan.xml` ' ta, MDB için yapılandırmada kullanmakta olduğunuz kuyruk yöneticisi ve kuyruk değerlerini ayarlayın.

WMQ_IVT_MDB ileti odaklı bean için:

1. queueManager öğesinin değerini, kuyruk yöneticinizin adı olacak şekilde ayarlayın.
2. Bir istemci bağlantısı kullanıyorsanız, çeşitli istemci bağlantı öğelerinin değerini, kuyruk yöneticinize yönelik bir bağlantı için uygun olacak şekilde ayarlayın.
3. Bağ tanımları bağlantısı kullanıyorsanız:
   a. transportType öğesinin değerini bağ tanımlarını (BATTACT) olarak ayarlayın.
   b. Çeşitli istemci bağlantısı öğelerini dışarı çıkarın ya da silin.
4. Değişikliklerinizi kaydedin.

Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması

IBM WebSphere MQ kaynak bağdaştırıcısını JBoss AS 5.1 ya da 6 olarak kurduktan sonra, kuruluş doğrulama sınaması (IVT) uygulamasını kurarak ve çalıştırarak kaynak bağdaştırıcısının kuruluşunu sınayarak sınayarak sınabilirsiniz.

Başlamadan önce

Önemli: Bu yönergeler, JBoss AS 5.1 ve 6 için geçerlidir, bunlar JBoss için 7 olarak geçerli değildir.

Kaynak bağdaştırıcısının JBoss EAP 6.3içine kurulmasıyla ilgili ek bilgi için bkz. “Kaynak bağdaştırıcısının JBoss EAP 6.3içinde kurulması ve sınanması” sayfa 759.

Bu görev, çalışmakta olan bir JBoss Server sunucusunun olduğunu ve bunun için standart yönetim görevlerini alıyorsanız, bu görevi devraldığını varsayar. Bu görev ayrıca, yerel sisteminizde bir IBM WebSphere MQ kuruluşuna sahip olduğunuzu ve standart yönetim görevlerini tanıdığınızı varsayar.

Bir IBM WebSphere MQ istemcisine bağlanmak için kaynak bağdaştırıcısı kullanıyorsanız ve dağıtılmış XA hareketleri gerekiyorsa, **Yalnızca istemci XA**olarak işaretlenmiş ek adımları izlemeniz gerekir. Bağ tanımlarına ve istemci kipi bağlantılarına ilişkin açıklamalar için bkz. "JMS için WebSphere MQ sınıflarına ilişkin bağlantı kipleri" sayfa 741.

Yordam

1. ExampleQMadlı bir kuyruk yöneticisi yaratın ve "Örnek programların hazırlanması ve çalıştırılması" sayfa 104içinde açıklandığı gibi ayarlayın.

   Kuyruk yöneticisini ayarlarken aşağıdaki noktalara dikkat edin:

   - Dinleyici, kapı 1414 'te başlatılmalıdır.
   - Kullanılacak kanalda SYSTEM.DEF.SVRCONN.
   - IVT uygulaması tarafından kullanılan kuyruk TEST.QUEUE.

   Ayrıca, bu uygulamanın geçici bir yanıt kuyruğu yaratabilmesi için, SYSTEM.DEFAULT.MODEL.QUEUE model kuyruğunun da DSP ve PUT yetkisi verilmesi gerekir.

   Farklı bir kuyruk yöneticisi, farklı bağlantı ayrıntıları ya da farklı bir kuyruk kullanmak istiyorsanız bkz. "Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması" sayfa 755.
2. Kaynak bağdaştırıcısı dosyasını (`wmq.jmsra.rar`), IVT uygulamasını (`wmq.jmsra.ivt.ear`) ve `jboss-jmsra-ds.xml` dosyasını edinin.

   Bu dosyaların konumu için bkz. " WebSphere MQ Resource Adapter ürününün kuruluşu" sayfa 705.
3. **Yalnızca İstemci XA**: Bağlantı üreticisine XA İşlemleri 'ni etkinleştirmek için `jboss-jmsra-ds.xml` dosyasını düzenleyin.
   a) Bağlantı üreticisi tanımlaması `<local-transaction/>`içindeki satırı açıklama satırı yapın ya da silin.
   b) Bağlantı üreticisi tanımlaması `<xa-transaction/>`içindeki satırı açıklama satırı olarak kaldırın.
   c) Değişikliklerinizi kaydedin.
4. **Yalnızca istemci XA**: (isteğe bağlı) İşlem gerektirmesi için MDB ' nin derleme tanımlayıcısını değiştirin. Bu işlem, IVT uygulaması bu değişiklik yapılmadan çalışmaya devam etse de, IVT 'deki MDB' yi XA hareketine katılmak üzere zorlar.
   a) `wmq.jmsra.ivt.ear` dosyasını açın.
   b) İçinde `WMQ_IVT_MDB.jar` dosyasını açın.
   c) Edit `META-INF/ejb-jar.xml`:
      i) Çevirme tanımlayıcısı içindeki satırı açıklama satırı yap ya da sil:

      ```
      <trans-attribute>NotSupported</trans-attribute>
      ```

      ii) Yapıbirimi tanımlayıcısı içindeki satırı açıklama satırı olarak kaldırın:

      ```
      <trans-attribute>Required</trans-attribute>
      ```

      iii) Değişikliklerinizi saklayın ve `WMQ_IVT_MDB.jar` dosyası içinde dosyayı güncelleyin.
      iv) Update the `wmq.jmsra.ivt.ear` file with the modified `WMQ_IVT_MDB.jar`.
5. Deploy the resource adapter to your server by copying the `wmq.jmsra.rar` file into the directory `jboss/server/default/deploy`.
6. Create the JMS resources required for the IVT Application by copying the `jboss-jmsra-ds.xml` file into the directory `jboss/server/default/deploy`.

7. Deploy the IVT application by copy the `wmq.jmsra.ivt.ear` file into the directory `jboss/server/default/deploy`.
8. IVT uygulamasını çalıştırın. Daha fazla ayrıntı için bkz. “ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750. JBossiçin varsayılan URL şudur: `http://localhost:8080/WMQ_IVT/`.

Deploying the IVT application in JBoss with a custom IBM WebSphere MQ environment

IBM WebSphere MQ kaynak bağdaştırıcısını JBossiçine kurarken, istemci kipi yerine farklı bir kuyruk, kuyruk yöneticisi, kapı, anasistem, kanal ya da bağ tanımları kipini kullanmak istiyorsanız, kaynak bağdaştırıcısı ya da IVT uygulamasını konuşlandırmadan önce JBoss ' ta IVT uygulamasını ve ilişkili komut dosyalarını değiştirmelisiniz.

Bu görev hakkında

Önemli: Bu yönergeler yalnızca, Java EE Sürüm 7 için değil, Java EE Sürüm 6 ve 5 için geçerlidir. Bu nedenle, JBoss Sürüm 8 (WildFly) için bu yönergelerin kullanımı desteklenmez.

“Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756' ta belirtilen farklı bir yapılandırmaya konuşlandırmak istiyorsanız, bu durumda, istemci kipi yerine farklı bir kuyruk yöneticisi, kuyruk, kapı, anasistem, kanal ya da bağ tanımları kipini kullanmak istiyorsanız, kaynak bağdaştırıcısı ya da IVT uygulamasını konuşlandırmadan önce aşağıdaki adımları tamamlayın.

Yordam

1. IVT uygulaması için kullanmak üzere farklı bir kuyruk yöneticisi ve kuyruk belirlemek istiyorsanız, kuyruk yöneticisi ve kuyruk için değer belirleyin.
   a) Jms/ivt/IVTCF bağlantı tanımı için:
      i) queueManager config-property değerini, kuyruk yöneticinizin adı olacak şekilde ayarlayın.
      ii) Bir istemci bağlantısı kullanıyorsanız, çeşitli istemci bağlantı öğelerinin değerini, kuyruk yöneticinize yönelik bir bağlantı için uygun olacak şekilde ayarlayın.
      iii) Bağ tanımları bağlantısı kullanıyorsanız, transportType öğesinin değerini BRETLER olarak ayarlayın ve daha sonra, çeşitli istemci bağlantısı öğelerini açıklama ya da silme işlemi için açıklama satırı yapın.
   b) Jms/iv/IVTQueue mbean için, baseQueueName öğesinin değerini, IVT uygulaması için yarattığınız kuyruğun adı olacak şekilde ayarlayın.
   c) Değişikliklerinizi kaydedin.
2. İletiyle yönlendirilen Bean (MDB) için yapılanıştaki farklı bir kuyruk yöneticisi ve kuyruk belirtmek istiyorsanız, MDB yapılanışını, kuyruk yöneticisine ve kuyruğa bağlanmaya ilişkin bir değişiklik yapmak üzere değiştirin.
   a) `wmq.jmsra.ivt.ear` dosyasını açın.
   b) İçinde `WMQ_IVT_MDB.jar` 'i açın.
   c) Edit `META-INF/ejb-jar.xml`:
      i) queueManager activation-config-property değerini, kuyruk yöneticinizin adı olacak şekilde ayarlayın.
      ii) Bir istemci bağlantısı kullanıyorsanız, çeşitli istemci bağlantısı etkinleştirme-config-özelliklerinin değerini, kuyruk yöneticinize yönelik bir bağlantı için uygun olacak şekilde ayarlayın.
      iii) Bir bağ tanımları bağlantısı kullanıyorsanız, transportType etkinleştirme-config-property özelliğinin değerini BRETLER olarak ayarlayın ve daha sonra, çeşitli istemci bağlantısı öğelerini açıklama ya da silme işlemi için açıklama satırı yapın ya da bağlantıyı silin.
   d) Değişiklikleri kaydedin ve `WMQ_IVT_MDB.jar` dosyası içinde dosyayı güncelleyin.
   e) Update the `wmq.jmsra.ivt.ear` file with the modified `WMQ_IVT_MDB.jar`.

3. Kaynak bağdaştırıcısını, bağ tanımları kipinde IBM WebSphere MQ ile bağlantı kurmak için yapılandırıyorsanız, JNI kitaplıklarının sistem yolunda olduğundan ya da JBossyolunun yolunda olduğundan emin olun. Ayrıntılar için bkz. “Java Native Interface (JNI) kitaplıklarının yapılandırılması” sayfa 696.

Sonraki adım

Continue deploying the resource adapter as described in “Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756.

.

Kaynak bağdaştırıcısının JBoss EAP 6.3içinde kurulması ve sınanması

After installing the IBM WebSphere MQ resource adapter in JBoss Enterprise Application Platform (EAP) 6.3, either on a standalone server or on a server running in a managed domain, you can test the installation of the resource adapter by installing and running the installation verification test (IVT) application.

Bu görev hakkında

Önemli: Bu yönergeler yalnızca JBoss EAP 6.3 için geçerli olur. Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulmasıyla ilgili ek bilgi için bkz. “Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756.

Bu görev, çalışmakta olan bir JBoss Server sunucusunun olduğunu ve bunun için standart yönetim görevlerini alıyorsanız, bu görevi devraldığını varsayar. Bu görev ayrıca, yerel sisteminizde bir IBM WebSphere MQ kuruluşuna sahip olduğunuzu ve standart denetimi tanıdığınızı varsayar.

Yordam

1. ExampleQMadlı bir kuyruk yöneticisi yaratın ve “Örnek programların hazırlanması ve çalıştırılması” sayfa 104içinde açıklandığı gibi ayarlayın.

   Kuyruk yöneticisini ayarlarken aşağıdaki noktalara dikkat edin:

   - Dinleyici, kapı 1414 'te başlatılmalıdır.
   - Kullanılacak kanalda SYSTEM.DEF.SVRCONN.
   - IVT uygulaması tarafından kullanılan kuyruk TEST.QUEUE.

   Ayrıca, bu uygulamanın geçici bir yanıt kuyruğu yaratabilmesi için, SYSTEM.DEFAULT.MODEL.QUEUE model kuyruğunun da DSP ve PUT yetkisi verilmesi gerekir.

   Farklı bir kuyruk yöneticisi, farklı bağlantı ayrıntıları ya da farklı bir kuyruk kullanmak istiyorsanız bkz. “Özel bir MQ ortamı ile IVT uygulamasının WAS CE üzerinde konuşlandırılması” sayfa 755.
2. Kaynak bağdaştırıcısı dosyasını (`wmq.jmsra.rar`) ve IVT uygulamasını (`wmq.jmsra.ivt.ear`) edinin.

   Bu dosyaların konumu için bkz. “ WebSphere MQ Resource Adapter ürününün kuruluşu” sayfa 705.
3. Kaynak bağdaştırıcısını takın ve kuruluş doğrulama sınaması (IVT) uygulamasını çalıştırarak kuruluşu test edin:

   - Kaynak bağdaştırıcısını bağımsız bir sunucuya kuruyorsanız bkz. “Bağımsız bir sunucuyu kurma ve test etme” sayfa 759.
   - Kaynak bağdaştırıcısını, yönetilen bir etki alanında çalışan bir sunucuya kuruyorsanız, “Yönetilen etki alanında çalışan bir sunucuyu kurma ve test etme” sayfa 761başlıklı konuya bakın.

Bağımsız bir sunucuyu kurma ve test etme

IBM WebSphere MQ kaynak bağdaştırıcısını bağımsız bir sunucuya JBoss EAP 6.3 üzerine kurduktan sonra, kuruluş doğrulama sınaması (IVT) uygulamasını kurarak ve çalıştırarak kaynak bağdaştırıcısının kuruluşunu sınayarak sınabilirsiniz.

Bu görev hakkında

Bu görevdeki bilgiler, kaynak bağdaştırıcısının bağımsız bir sunucuya kurulması ve sınanması içindir. Kaynak bağdaştırıcısını, yönetilen bir etki alanında çalışan bir sunucuya kuruyorsanız, “Yönetilen etki alanında çalışan bir sunucuyu kurma ve test etme” sayfa 761başlıklı konuya bakın.

Önemli: Bu yönergeler yalnızca JBoss EAP 6.3 için geçerli olur. Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulmasıyla ilgili ek bilgi için bkz. “Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756.

Yordam

1. Deploy the resource adapter to your server by copying the `wmq.jmsra.rar` file into the directory <*EAP_HOME*>`/standalone/deployments`.
2. Create the JMS resources required for the IVT Application by adding the following entries to the `<resource-adapters>` section of the <*EAP_HOME*>`/standalone/configuration/standalone-full.xml` file:

```
<resource-adapter id="wmq.jmsra">
  <archive>
      wmq.jmsra.rar
  </archive>
  <transaction-support>NoTransaction</transaction-support>
  <connection-definitions>
     <connection-definition
          class-name="com.ibm.mq.connector.outbound.ManagedConnectionFactoryImpl"
          jndi-name="java:jboss/jms/ivt/IVTCF"
          enabled="true"
          use-java-context="true"
          pool-name="IVTCF">
       <config-property name="port">
           1414
       </config-property>
       <config-property name="hostName">
           localhost
       </config-property>
       <config-property name="channel">
           SYSTEM.DEF.SVRCONN
       </config-property>
       <config-property name="transportType">
           CLIENT
       </config-property>
       <config-property name="queueManager">
           ExampleQM
       </config-property>
     </connection-definition>
  </connection-definitions>
  <admin-objects>
    <admin-object class-name="com.ibm.mq.connector.outbound.MQQueueProxy"
                  jndi-name="java:jboss/jms/ivt/IVTQueue"
                  pool-name="IVTQueue">
      <config-property name="baseQueueName">
          TEST.QUEUE
      </config-property>
    </admin-object>
  </admin-objects>
</resource-adapter>
```

3. Uygulama sunucunuzun başlatma değiştirgelerine aşağıdaki bilgileri ekleyin:

```
-Dcom.ibm.mq.connector.IVTMDBCFJNDIName=java:jboss/jms/ivt/IVTCF
```

4. Deploy the IVT application by copying the `wmq.jmsra.ivt.ear` file into the directory <*EAP_HOME*>`/standalone/deployments`.
5. Uygulama sunucusunu başlatın.
6. IVT uygulamasını çalıştırın.

   Daha fazla bilgi için, bkz. “ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750. JBossiçin varsayılan URL şudur: `http://localhost:8080/WMQ_IVT/`.

Not: IVT uygulamasını çalıştırmak için gereken JMS kaynakları için kullanılan JNDI adları aşağıdaki gibidir:

```
Connection Factory : IVTCF
JNDI name          : java:jboss/jms/ivt/IVTCF

Destination        : IVTQueue
JNDI name          : java:jboss/jms/ivt/IVTQueue
```

Yukarıda belirtilen URL ' yi kullanarak IVT uygulamasını başlattığınızda, ilgili alanlara bu kaynakların JNDI adlarını girin ve **IVT Çalıştır**düğmesini tıklatarak uygulamayı çalıştırın.

Yönetilen etki alanında çalışan bir sunucuyu kurma ve test etme

IBM WebSphere MQ kaynak bağdaştırıcısını, yönetilen bir etki alanında çalışan bir sunucuya JBoss EAP 6.3 üzerine kurduktan sonra, kuruluş doğrulama sınaması (IVT) uygulamasını kurarak ve çalıştırarak kaynak bağdaştırıcısının kuruluşunu sınayarak sınabilirsiniz.

Bu görev hakkında

Bu görevdeki bilgiler, yönetilen bir etki alanında çalışan bir sunucuya kaynak bağdaştırıcısının kurulması ve sınanması içindir. Kaynak bağdaştırıcısını bağımsız bir sunucuya kuruyorsanız bkz. “Bağımsız bir sunucuyu kurma ve test etme” sayfa 759.

Önemli: Bu yönergeler yalnızca JBoss EAP 6.3 için geçerli olur. Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulmasıyla ilgili ek bilgi için bkz. “Kaynak bağdaştırıcısının JBoss AS 5.1 ve 6 olarak kurulması ve sınanması” sayfa 756.

Yordam

1. JBoss Yönetim Konsolu ya da Yönetim CLI 'sını kullanarak kaynak bağdaştırıcısını sunucunuza yerleştirin.
2. Create the JMS resources required for the IVT Application by adding the following entries to the <resource-adapters> section of the <EAP_HOME>/domain/configuration/domain.xml file:

```
<resource-adapter id="wmq.jmsra">
  <archive>
      wmq.jmsra.rar
  </archive>
  <transaction-support>NoTransaction</transaction-support>
  <connection-definitions>
    <connection-definition
         class-name="com.ibm.mq.connector.outbound.ManagedConnectionFactoryImpl"
         jndi-name="java:jboss/jms/ivt/IVTCF"
         enabled="true"
         use-java-context="true"
         pool-name="IVTCF">
      <config-property name="port">
          1414
      </config-property>
      <config-property name="hostName">
          localhost
      </config-property>
      <config-property name="channel">
          SYSTEM.DEF.SVRCONN
      </config-property>
      <config-property name="transportType">
          CLIENT
      </config-property>
      <config-property name="queueManager">
          ExampleQM
      </config-property>
    </connection-definition>
  </connection-definitions>
  <admin-objects>
    <admin-object class-name="com.ibm.mq.connector.outbound.MQQueueProxy"
                  jndi-name="java:jboss/jms/ivt/IVTQueue"
                  pool-name="IVTQueue">
      <config-property name="baseQueueName">
          TEST.QUEUE
```

```
      </config-property>
    </admin-object>
  </admin-objects>
</resource-adapter>
```

3. Uygulama sunucunuzun başlatma değiştirgelerine aşağıdaki bilgileri ekleyin:

```
-Dcom.ibm.mq.connector.IVTMDBCFJNDIName=java:jboss/jms/ivt/IVTCF
```

4. Uygulama sunucusunu durdurup yeniden başlatın.
5. Deploy the IVT application using the JBoss Management Console or Management CLI.
6. IVT uygulamasını çalıştırın.

   Daha fazla bilgi için, bkz. “ WebSphere MQ kaynak bağdaştırıcısına ilişkin kuruluş doğrulama sınaması programı” sayfa 750. JBossiçin varsayılan URL şudur: `http://localhost:8080/WMQ_IVT/`.

   Not: IVT uygulamasını çalıştırmak için gereken JMS kaynakları için kullanılan JNDI adları aşağıdaki gibidir:

```
Connection Factory : IVTCF
JNDI name          : java:jboss/jms/ivt/IVTCF

Destination        : IVTQueue
JNDI name          : java:jboss/jms/ivt/IVTQueue
```

   Yukarıda belirtilen URL 'yi kullanarak IVT uygulamasını başlattığınızda ve daha sonra, **IVT Çalıştır**' ı tıklatarak uygulamayı çalıştırdığınızda, bu kaynakların JNDI adlarını kendi alanlarına girin.

JMS için WebSphere MQ sınıflarıyla sağlanan komut dosyaları

JMS için WebSphere MQ sınıfları kullanılırken gerçekleştirilmesi gereken genel görevlere yardımcı olmak için bir dizi komut dosyası sağlanmıştır.

Çizelge 102 sayfa 762 , tüm komut dosyalarını ve bunların kullanımlarını listeler. Komut dosyaları, JMS kuruluş dizini için WebSphere MQ sınıflarının `bin` alt dizininde bulunur.

Çizelge 102. JMS için WebSphere MQ sınıflarıyla sağlanan komut dosyaları

Yardımcı Program	**Kullan**
Temizleme[1]	Bu komut dosyası önceki yayınlarla uyumluluk için korunur, ancak hiçbir işlev gerçekleştirmez. Abonelik bilgilerini el ile temizleme işlemi artık gerekli değil
DefaultConfiguration	Varsayılan yapılandırma uygulamasını Windowsdışındaki platformlarda çalıştırır.
formatlog[1]	Bu komut dosyası önceki yayınlarla uyumluluk için korunur, ancak hiçbir işlev gerçekleştirmez. Günlük çıkışı artık okunabilir metinde üretildi.
IVTRun[1] IVTSetup[1] VTTtiry[1]	Noktadan noktaya kuruluş doğrulama sınamasında (“JMS için WebSphere MQ sınıfları için noktadan noktaya kuruluş doğrulama sınaması” sayfa 743) açıklandığı gibi kullanılır.
JMSAdmin[1]	Runs the WebSphere MQ JMS administration tool, as described in “IBM WebSphere MQ classes for JMS denetim aracını çağırma” sayfa 896.
JMSAdmin.config	WebSphere MQ JMS yönetim aracına ilişkin yapılandırma dosyası (“JMS Denetimi aracının yapılandırılması” sayfa 897içinde açıklandığı gibi).

Çizelge 102. JMS için WebSphere MQ sınıflarıyla sağlanan komut dosyaları (devamı var)	
Yardımcı Program	**Kullan**
PSIVTRun[1]	Runs the publish/subscribe installation verification test program, as described in “JMS için WebSphere MQ sınıfları için yayınlama/abone olma kuruluş doğrulama sınaması” sayfa 746.
PSReportDump.class	Bu sınıf önceki yayın düzeyleriyle uyumluluk için korunur, ancak hiçbir işlev gerçekleştirmez.
setjmsenv	Sets the environment variables for running a WebSphere MQ classes for JMS application in a 32-bit Java virtual machine (JVM) on UNIX and Linux systems, as described in “JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri” sayfa 694.
setjmsenv64	Sets the environment variables for running a WebSphere MQ classes for JMS application in a 64-bit JVM on UNIX and Linux systems, as described in “JMS için IBM WebSphere MQ sınıfları tarafından kullanılan ortam değişkenleri” sayfa 694.
Not: 1. Windowsüzerinde, dosya adı .bat uzantısına sahiptir.	

OSGi desteği

OSGi, uygulamaların kod paketleri olarak konuşlandırılmasını destekleyen bir çerçeve sağlar. Dokuz OSGi kod paketi IBM WebSphere MQ classes for JMS ' nin bir parçası olarak sağlanır.

OSGi, kod paketleri biçiminde gelen uygulamaların konuşlandırılmasını destekleyen genel amaçlı, güvenli ve yönetilenJava çerçevesini sağlar. OSGi uyumlu aygıtlar, paketleri yükleyebilir ve kurabilir ve artık gerekli olmadığında bunları kaldırabilir. Çerçeve, kod paketlerinin dinamik ve ölçeklenebilir bir şekilde kurulup güncellenmesini yönetir.

IBM WebSphere MQ classes for JMS. Aşağıdaki OSGi kod paketlerini içerir.

com.ibm.msg.client.osgi.jms< sürüm numarası > .jar
IBM WebSphere MQ classes for JMSiçindeki ortak kod katmanı. JMS için WebSphere MQ sınıflarının katmanlı mimarisine ilişkin bilgi için bkz. “Katmanlı bir mimari” sayfa 766.

com.ibm.msg.client.osgi.jms.prereq_< sürüm numarası > .jar
Ortak katmana ilişkin önkoşul Java arşiv (JAR) dosyaları.

com.ibm.msg.client.osgi.commonservices.j2se_<version sayı > .jar
Common services for Java Platform, Standard Edition (Java SE) applications.

com.ibm.msg.client.osgi.nls_< sürüm numarası > .jar
Ortak katmana ilişkin iletiler.

com.ibm.msg.client.osgi.wmq_< sürüm numarası > .jar
IBM WebSphere MQ classes for JMSiçindeki IBM WebSphere MQ ileti alışverişi sağlayıcısı. For information about the layered architecture of IBM WebSphere MQ classes for JMS , see “Katmanlı bir mimari” sayfa 766.

com.ibm.msg.client.osgi.wmq.prereq_< sürüm numarası > .jar
IBM WebSphere MQ ileti alışverişi sağlayıcısına ilişkin önkoşul JAR dosyaları.

com.ibm.msg.client.osgi.wmq.nls_< sürüm numarası > .jar
IBM WebSphere MQ ileti alışverişi sağlayıcısına ilişkin iletiler.

com.ibm.mq.osgi.directip_< sürüm numarası > .jar
IBM WebSphere MQ ileti alışverişi sağlayıcısının bir aracıya gerçek zamanlı bağlantı yaratmasına olanak tanıyan JAR dosyaları.

Burada < sürüm numarası >, kurulu olan WebSphere MQ ' un sürüm sayısıdır.

Bu paketler, WebSphere MQ kurulumunuzun `java/lib/OSGi` alt dizinine ya da Windows 'ta `java\lib\OSGi` klasörüne kurulur.

The bundle com.ibm.mq.osgi.java <version number>.jar, which is also installed into the `java/lib/OSGi` subdirectory of your WebSphere MQ installation, or the `java\lib\OSGi` folder on Windows, is part of the WebSphere MQ classes for Java. Bu kod paketi, JMS için WebSphere MQ sınıflarına sahip bir OSGi çalıştırma zamanı ortamına yüklenmemelidir.

JMS için WebSphere MQ sınıflarına ilişkin OSGi paketleri OSGi Yayın Düzeyi 4 belirtimine yazıldı. Bir OSGi Yayın Düzeyi 3 ortamında çalışmıyorlar.

OSGi yürütme ortamı gereken DLL kütüklerini ya da paylaşılan kitaplıkları bulabilmesi için, sistem yolunu ya da kitaplık yolunu doğru olarak ayarlamalısınız.

IBM WebSphere MQ classes for JMSiçin OSGi paketlerini kullanıyorsanız, geçici konular işe yaramaz. Ayrıca, OSGi gibi birden çok sınıf yükleyici ortamındaki sınıfların yüklenmesi sırasında sorunlu bir sorun nedeniyle Java içinde yazılan kanal çıkış sınıfları desteklenmez. Bir kullanıcı paketi, JMS paketleri için IBM WebSphere MQ sınıflarından haberdar olabilir, ancak IBM WebSphere MQ classes for JMS kod paketleri herhangi bir kullanıcı kod paketinin farkında değildir. Sonuç olarak, bir IBM WebSphere MQ classes for JMS kod paketinde kullanılan sınıf yükleyici, bir kullanıcı kod paketinde bulunan bir kanal çıkış sınıfını yükleyemez.

OSGi hakkında daha fazla bilgi için OSGi Alliance web sitesine bakın.

JMS için IBM WebSphere MQ sınıflarıyla ilgili sorunların çözülmesi

Kuruluş doğrulama programlarını çalıştırabilir ve izleme ve günlük olanaklarını kullanarak sorunları araştırabilirsiniz.

If a program does not complete successfully, run one of the installation verification programs, as described in “JMS için WebSphere MQ sınıfları için noktadan noktaya kuruluş doğrulama sınaması” sayfa 743 and “JMS için WebSphere MQ sınıfları için yayınlama/abone olma kuruluş doğrulama sınaması” sayfa 746, and follow the advice given in the diagnostic messages.

Günlüğe kaydetme ve IBM WebSphere MQ classes for JMS

Varsayılan olarak, günlük çıkışı `mqjms.log` dosyasına gönderilir. Dosyayı belirli bir dosyaya ya da dizine yönlendirebilirsiniz.

The IBM WebSphere MQ classes for JMS log facility is provided to report serious problems, particularly problems that might indicate configuration errors rather than programming errors. Varsayılan olarak, günlük çıkışı JVM çalışma dizinindeki `mqjms.log` dosyasına gönderilir.

You can redirect log output to another file by setting the property com.ibm.msg.client.commonservices.log.outputName. Bu özelliğin değeri şunlar olabilir:

- Tek bir yol adı.
- Yol adlarının virgülle ayrılmış bir listesi (tüm veriler için tüm veriler günlüğe kaydedilir).

Her yol adı şöyle olabilir:

- Mutlak ya da göreli.
- Standart hata akışını temsil eden`stderr` ya da `System.err` .
- Standart çıkış akımını göstermek için`sttdout` ya da `System.out` .

Özelliğin değeri bir dizini tanımlıyorsa, günlük çıkışı bu dizinde `mqjms.log` değerine yazılır. Özelliğin değeri belirli bir dosyayı tanımlıyorsa, günlük çıkışı o dosyaya yazılır.

Bu özelliği IBM WebSphere MQ classes for JMS yapılandırma dosyasında ya da **`java`** komutunda bir sistem özelliği olarak ayarlayabilirsiniz. Aşağıdaki örnekte, özellik bir sistem özelliği olarak ayarlanmıştır ve belirli bir dosyayı tanımlar:

```
java -Djava.library.path=library_path
     -Dcom.ibm.msg.client.commonservices.log.outputName=/mydir/mylog.txt
      MyAppClass
```

Komutta *library_path* , IBM WebSphere MQ classes for JMS kitaplıklarını içeren dizinin yoludur (bkz. “Java Native Interface (JNI) kitaplıklarının yapılandırılması” sayfa 696).

com.ibm.msg.client.commonservices.log.status özelliğini OFF değerine ayarlayarak günlük çıkışını geçersiz kılabilirsiniz. Bu özelliğin varsayılan değeri ON (AÇIK) değeridir.

`System.err` ve `System.out` değerleri, günlük çıkışını `System.err` ve `System.out` akışlarına göndermek için ayarlanabilir.

JMS için WebSphere MQ sınıflarına giriş, programcılar için

JMS için WebSphere MQ sınıfları, WebSphere MQile birlikte verilen JMS sağlayıcısıdır. JMS için WebSphere MQ sınıfları, javax.jms paketinde tanımlanan arabirimleri uygular ve aynı zamanda JMS API ' ya iki uzantı kümesi sağlar. Java Platform, Standard Edition (Java SE) ve Java Platform, Enterprise Edition (Java EE) uygulamaları, JMS için WebSphere MQ sınıflarını kullanabilir.

JMS belirtimi, uygulamaların ileti alışverişi işlemlerini gerçekleştirmek için kullanabilecekleri bir arabirim kümesini tanımlar. Belirtimin en son sürümü, Sürüm 1.1' dir. javax.jms paketi, JMS arabirimlerinin ayrıntılarını belirtir ve bir JMS sağlayıcısı, belirli bir ileti alışverişi ürünü için bu arabirimleri uygular. JMS için WebSphere MQ sınıfları, WebSphere MQiçin JMS arabirimlerini uygulayan bir JMS sağlayıcısıdır.

Bir JMS uygulaması içindeki mantık akışı, ConnectionFactory ve Destination nesneleriyle başlar. Uygulama, uygulamadan bir ileti alışverişi sunucusuna etkin bağlantıyı gösteren bir Connection nesnesi yaratmak için bir ConnectionFactory nesnesi kullanır. Uygulama, iletiler üretmek ve tüketmek için tek bir iş parçacıklı bağlam olan bir Oturum nesnesi yaratmak için Connection nesnesini kullanır. Uygulama daha sonra, uygulamanın belirtilen hedefe ileti göndermek için kullandığı bir MessageProducer nesnesi yaratmak için Oturum nesnesini ve hedef nesneyi kullanabilir. Hedef, ileti sistemi sistemindeki bir kuyrukta ya da bir konudur ve Hedef nesne tarafından kapsüllenmiş olur. Uygulama ayrıca, uygulamanın belirtilen hedefe gönderilen iletileri almak için kullandığı bir MessageConsumer nesnesi yaratmak için Oturum nesnesini ve hedef nesneyi kullanabilir.

JMS belirtimi, ConnectionFactory ve Hedef nesnelerin denetlenmesini bekler. Bir denetimci, denetlenen nesneleri merkezi bir havuzda yaratır ve bakımını yapar; JMS uygulaması bu nesneleri Java Naming and Directory Interface (JNDI) kullanarak alır. Denetlenen nesnelerin havuzu, basit bir dosyadan bir LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizinine kadar aralığa neden olabilir.

JMS için WebSphere MQ sınıfları, denetimli nesne kullanımını destekler. An application can use all the features of WebSphere MQ that are exposed through WebSphere MQ classes for JMS without having any WebSphere MQ-specific information hard coded into the application itself. Bu düzenleme, temeldeki WebSphere MQ yapılanışından bir derece bağımsızlığa sahip bir uygulama sağlar. Bu bağımsızlığı elde etmek için uygulama, yönetilen nesneler olarak depolanan bağlantı fabrikalarını ve hedeflerini almak için JNDI kullanabilir ve ileti alışverişi işlemlerini gerçekleştirmek için yalnızca javax.jms paketinde tanımlanan arabirimleri kullanır. An administrator can use the WebSphere MQ JMS administration tool or IBM WebSphere MQ Explorer to create and maintain administered objects in a central repository. Ancak, bir uygulama sunucusu, nesneleri oluşturmak ve korumak için yönetilen nesneler ve kendi araçları için kendi havuzunu sağlar. Bu nedenle bir Java EE uygulaması, uygulama sunucusu havuzundan ya da merkezi bir havuzdan yönetilen nesneleri almak için JNDI kullanabilir.

JMS için WebSphere MQ sınıfları da JMS API ' ye uzantılar sağlar. JMS için önceki WebSphere MQ sınıflarının önceki yayın düzeyleri, MQConnectionFactory, MQQueue ve MQTopic nesnelerinde uygulanan uzantıları içerir. Bu nesneler, WebSphere MQ' e özgü özellikleri ve yöntemlere sahiptir. Nesneler yönetilebilir ya da bir uygulama, çalıştırma zamanında nesneleri dinamik olarak yaratabilir. JMS için WebSphere MQ sınıflarının bu yayın düzeyi, bu uzantıların bakımını yapar ve bu uzantıları kullanan uygulamalar, değişiklik olmadan, kullanmaya devam edebilirsiniz. Bu uzantılar, *WebSphere MQ JMS uzantıları*olarak bilinir. Bu belge kümesinde, yürütme sırasında bir uygulama tarafından devingen olarak yaratılan nesneler, denetlenen nesneler olarak *dikkate alınmadığından* dikkat çekmez.

In addition to the WebSphere MQ JMS extensions, this release of WebSphere MQ classes for JMS provides a more generic set of extensions to the JMS API. Bu uzantılar, *IBM JMS uzantıları*olarak bilinir ve aşağıdaki geniş hedeflere sahiptir:

- IBM JMS sağlayıcıları arasında daha yüksek bir tutarlılık düzeyi sağlamak için
- İki IBM ileti sistemi sistemi arasında bir köprü uygulaması yazmanın daha kolay olmasını sağlamak için
- Bir uygulamanın bir IBM JMS sağlayıcısından başka bir uygulamayı daha kolay kapmasını sağlamak için

Bu uzantıların ana odağı, yürütme sırasında bağlantı üreticilerinin ve hedeflerin dinamik olarak oluşturulması ve yapılandırılması ile ilgilidir; ancak, uzantılar sorunun saptanması için işlev gibi ileti sistemiyle doğrudan ilgili olmayan bir işlev de sağlar.

javax.jms arabirimi ya da JMS uzantıları kümesi kullanılarak oluşturulan bir bağlantı üreticisi, kuyruğu ya da konu nesnesi bu API'lardan herhangi biri kullanılarak adreslenilebilir; bu, arabirimlerin herhangi birine dönüştürülebilmektedir. Uygulama taşınabilirliğini en üst düzeyde tutmak için gereksinimlerinize uygun en genel API ' yı kullanın.

Hem Java SE hem de Java EE uygulamaları, JMS için WebSphere MQ sınıflarını kullanabilir. On the Java EE platform, WebSphere MQ classes for JMS supports two types of communication between a component of an application and a WebSphere MQ queue manager:

Giden iletişim
JMS API ' yı doğrudan kullanarak, bir uygulama bileşeni kuyruk yöneticisine bağlantı yaratır ve iletileri gönderir ve alır.

Örneğin, uygulama bileşeni bir uygulama istemcisi, bir sunucu uygulamacığı, bir JavaServer Sayfası (JSP), kurumsal Java Bean (EJB) ya da ileti odaklı bir Bean (MDB) olabilir. Bu iletişim tipinde, uygulama sunucusu taşıyıcısı, bağlantı havuzlama ve iş parçacığı yönetimi gibi ileti alışverişi işlemlerini desteklemek için yalnızca düşük düzeyli işlevler sağlar.

Gelen iletişim
Bir hedefe gelen bir ileti bir MDB ' ye teslim edilir ve sonra iletiyi işler.

Java EE uygulamaları, iletileri zamanuyumsuz olarak işlemek için MDBS ' yi kullanır. Bir MDB, JMS ileti dinleyicisi olarak işlev görür ve bir iletinin nasıl işleneceğini tanımlayan bir onMessage() yöntemi tarafından uygulanmaktadır. Bir MDB, bir uygulama sunucusunun EJB taşıyıcısında konuşlandırılır. Bir MDB 'nin yapılandırıldığı kesin yol, kullandığınız uygulama sunucusuna bağlıdır; ancak, yapılanış bilgilerinin hangi kuyruk yöneticisinin bağlanacağını, kuyruk yöneticisine nasıl bağlanacağını, hangi hedefe ilişkin iletileri izleyeceğini ve MDB' nin hareket davranışını belirtmesi gerekir. Bu bilgi daha sonra EJB taşıyıcısı tarafından kullanılır. Belirtilen hedefe MDB 'nin seçim ölçütlerini karşılayan bir ileti geldiğinde, EJB taşıyıcısı iletiyi kuyruk yöneticisinden almak için JMS için WebSphere MQ sınıflarını kullanır ve daha sonra, onMessage() yöntemini çağırarak iletiyi MDB' ye aktarır.

JMS mimarisine ilişkinIBM WebSphere MQ sınıfları

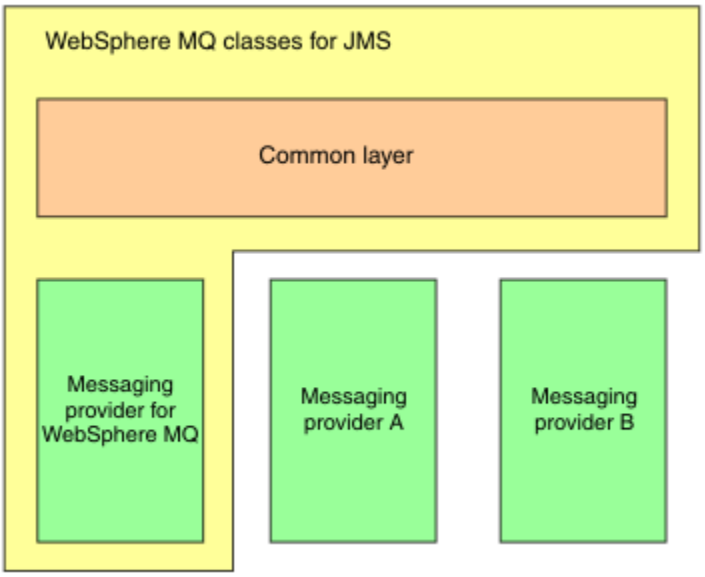
JMS içinIBM WebSphere MQ sınıfları, IBM WebSphere MQ Sürüm 7.0' de sağlanan ve sonraki yayınlarda, önceki yayınlarla karşılaştırıldığında bir dizi geliştirmeyi içerir. Bu geliştirmelerin bazıları, JMS için IBM WebSphere MQ sınıflarının uygulanmasında yapılan değişikliklerin bir sonucu olarak ve bazıları, temel IBM WebSphere MQ işlevinde yapılan değişiklikleri JMS kullanan JMS için IBM WebSphere MQ sınıflarının bir sonucu olarak yer alıyor.

Aşağıdaki kısımlarda temel geliştirmeler özetlenmektedir.

Katmanlı bir mimari

Önceki WebSphere MQyayınlarında, JMS için WebSphere MQ sınıflarının somutlaması tamamen WebSphere MQ' a özgülemiştir. İleti sistemi sistemleri sağlayan diğer IBM ürünleri de JMS sağlayıcılarını içerir, ancak bu JMS sağlayıcılar çok az ya da JMS için WebSphere MQ sınıflarının uygulanmasıyla ortak bir ortak hiçbir şey içermez.

JMS için WebSphere MQ V7.0, WebSphere MQ sınıflarından katmanlı bir mimari vardır. En üst kod katmanı, herhangi bir IBM JMS sağlayıcısı tarafından kullanılabilen ortak bir katmanıdır. Bir uygulama bir JMS yöntemini çağırdığında, bir ileti sistemi sistemine özgü olmayan bir çağrı işlemi, çağrıya tutarlı bir yanıt da sağlayan ortak katman tarafından gerçekleştirilir. Bir ileti sistemi sistemine özgü çağrıların işlenmesi daha düşük bir katmana devredilir. Şekil 125 sayfa 767 , katmanlı mimariyi gösterir.



Şekil 125. IBM JMS sağlayıcılar için katmanlı mimari

Katmanlı bir mimariye taşınmak, aşağıdaki hedefleri içerir:

- Çeşitli IBM JMS sağlayıcılarının davranışlarının tutarlılığını artırmak için
- İki IBM ileti sistemi sistemi arasında bir köprü uygulaması yazmanın daha kolay olmasını sağlamak için
- Bir uygulamanın bir IBM JMS sağlayıcısından başka bir uygulamayı daha kolay kapmasını sağlamak için

JMS için WebSphere MQ sınıflarının bu somutlaması, JMS API ' ya yeni bir uzantı kümesi de sunar. Bu uzantılar, *IBM JMS uzantıları*olarak bilinir. Bu uzantıların ana odağı, yürütme sırasında bağlantı üreticilerinin ve hedeflerin dinamik olarak oluşturulması ve yapılandırılmasına ilişkin endişeler ile ilgilenir.

IBM JMS uzantılarını kullanan bir uygulama, seçilen ileti sistemi sistemini tanımlayan bir sabiti değiştirge olarak belirterek bir JmsFactoryFactory nesnesi yaratılarak başlar. Uygulama, seçilen ileti sistemi için doğru özel sınıflara sahip bağlantı fabrikaları ve hedefler yaratmak için JmsFactoryFactory nesnesini kullanır.

Uygulama, daha sonra, bağlantı fabrikalarını ve hedeflerini ayarlayarak özelliklerini tanımlayabilir. IBM JMS uzantıları, özellikleri ayarlamak için bir yöntem kümesi sağlar. Bu yöntemler, herhangi bir ileti sistemi sisteminden bağımsızdır. Her veri tipinin kendine özgü bir set yöntemi vardır ve her özellik, WMQConstants sınıfının durağan son üyesi olarak tanımlanan bir adla tanıtılır. Bir uygulama bu yöntemlerden birini çağırdığında, çağrıdaki değiştirgelerden biri özelliğin adı, diğer değiştirge ise özelliğin değeridir.

For example, if WebSphere MQ is the messaging system, one of the properties of a connection factory is the name of the queue manager to connect to. Bir uygulama, IBM JMS uzantılarını kullanarak, aşağıdaki yöntemi çağırarak kuyruk yöneticisinin adını JÜPITER 'e ayarlar:

```
JmsConnectionFactory myCF;
...
myCF.setStringProperty(WMQConstants.WMQ_QUEUE_MANAGER, "JUPITER");
```

Buna karşılık, uygulama aşağıdaki yöntemi çağırarak aynı işlevi gerçekleştirebilir:

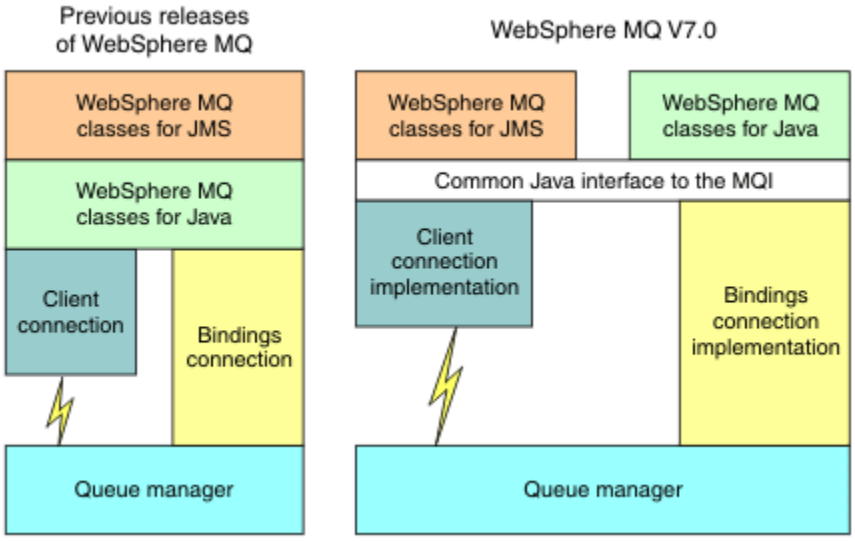
```
MQConnectionFactory myCF;
...
myCF.setQueueManager("JUPITER");
```

Bu yöntem, bir WebSphere MQ JMS uzantısıdır ve ileti sistemi sistemi olarak WebSphere MQ ' ya özgüdür. Bu nedenle, bu yöntemin kullanılması, uygulamanın daha kolay bir şekilde başka bir IBM JMS sağlayıcısına daha kolay bir bağlantı noktası haline getirmesini sağlar.

JMS için WebSphere MQ sınıfları ve Java için WebSphere MQ sınıfları arasındaki ilişki

In releases of WebSphere MQ, prior to Version 7.0, WebSphere MQ classes for JMS was implemented almost entirely as a layer of code on top of WebSphere MQ classes for Java . Bu düzenleme, uygulama geliştiricileri arasında bazı karışıklığa neden olmuştur; çünkü, MQEnvironment sınıfında alanlar ya da çağrılan yöntemler, JMS için WebSphere MQ sınıfları kullanılarak yazılmış kodun yürütme zamanı davranışında istenmeyen ve beklenmeyen etkilere neden olabilir. Buna ek olarak, JMS için WebSphere MQ sınıflarının uygulanması, JMS API ' nin Java için WebSphere MQ sınıflarının üstüne sığmadığı ve bu kısıtlamaların yürütme zamanı performansıyla ilgili bazı sorunlara yol açmadığı alanlarda bazı kısıtlar ortaya konuyor.

WebSphere MQ V7.0' den, JMS için WebSphere MQ sınıflarının uygulaması artık Java için WebSphere MQ sınıflarına bağımlı değildir. JMS için WebSphere MQ sınıfları ve JMS için WebSphere MQ sınıfları artık MQI ' ye ortak bir Java arabirimi kullanan eşdüzeylerdir. Bu düzenleme, performansın en iyi duruma getirilmesi için daha fazla kapsama olanak sağlar ve MQEnvironment sınıfındaki alanların ya da arama yöntemlerinin, JMS için WebSphere MQ sınıfları kullanılarak yazılan kod yürütme davranışı üzerinde hiçbir etkisi olmadığını gösterir. Şekil 126 sayfa 768 shows the relationship between WebSphere MQ classes for JMS and WebSphere MQ classes for Java in previous releases of WebSphere MQ and in WebSphere MQ V7.0 and subsequent releases.



Şekil 126. JMS için WebSphere MQ sınıfları ve Java için WebSphere MQ sınıfları arasındaki ilişki

In order to maintain compatibility with earlier releases, channel exit classes written in Java can still use the WebSphere MQ classes for Java interfaces, even if the channel exit classes are called from WebSphere MQ classes for JMS. Ancak, Java arabirimleri için WebSphere MQ sınıflarının kullanılması, uygulamalarınızın hala Java JAR dosyası com.ibm.mq.jariçin WebSphere MQ sınıflarına bağımlı olduğu anlamına gelir. Sınıf yolunuzda com.ibm.mq.jar öğesini istemiyorsanız, bunun yerine com.ibm.mq.exits paketindeki yeni arabirim takımını kullanabilirsiniz.

You can now create and configure JMS administered objects with the WebSphere MQ Explorer.

Yayınlama/abone olma ileti alışverişi

WebSphere MQ V7.0ve sonraki yayınlar, gömülü yayınlama/abone olma işlevini içerir. Bu işlev, WebSphere MQ V6.0ile sağlanan WebSphere MQ Publish/Subscribe (Yayınlama/Abone Olma) öğesini değiştirir.

JMS uygulamaları için WebSphere MQ sınıfları, gömülü yayınlama/abone olma işlevini kullanabilir ve iletim olarak WebSphere MQ ile ileti alışverişi için WebSphere Event Broker ya da WebSphere Message Broker olanağını kullanmak yerine, bu işlevi kullanabilir. Yeni işlevi kullanmak üzere JMS için WebSphere MQ sınıflarının yapılandırılması, JMS için WebSphere MQ sınıflarının WebSphere MQ Publish/Subscribe,

WebSphere Event Broker ya da WebSphere Message Broker olanağını kullanması için yapılandırılmaktan daha basittir. Yöneticilerin ve uygulama geliştiricilerinin artık yayın kuyruklarını, abone kuyruklarını, abonelik depolarını ve abone temizliğini yönetmeye gerek kalmaması gerekir. Ayrıca, ConnectionFactory ve Konu nesnelerinde daha az sayıda özellik vardır.

Yerleşik yayınlama/abone olma işlevi, bir uygulamanın abone olmak istediği bir dizi konu aralığını belirtmek için alıkonalan yayınlar ve iki genel arama tipi şemaları seçeneği gibi bazı ek özellikler de sağlar.

An application can still use a real-time connection to a broker of WebSphere Event Broker or WebSphere Message Broker for publish/subscribe messaging. Bu destek değiştirilmez.

Applications using WebSphere MQ Publish/Subscribe can use the embedded publish/subscribe function without change when the queue manager to which they are connected is upgraded. Bir uygulama tarafından ayarlanan, ancak gömülü yayınlama/abone olma işleviyle zorunlu olmayan özellikler yoksayılır.

WebSphere MQ ileti sistemi sağlayıcısı

WebSphere MQ ileti alışverişi sağlayıcısında iki işlem kipi vardır:

- *WebSphere MQ ileti alışverişi sağlayıcısı olağan kipi*
- *WebSphere MQ Messaging Provider geçiş kipi*

The WebSphere MQ messaging provider normal mode uses all the features of the WebSphere MQ Version 7.0, and subsequent release queue managers to implement JMS. Bu kip yalnızca bir WebSphere MQ kuyruk yöneticisine bağlanmak için kullanılır ve her iki istemci ya da bağ tanımlama kipinde WebSphere MQ Sürüm 7.0ve sonraki yayın kuyruğu yöneticilerine bağlanabilirler. Bu kip, yeni WebSphere MQ Sürüm 7.0 ve sonraki yayın işlevini kullanmak için eniyilenir.

WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipi, WebSphere MQ Sürüm 6.0 işlevini temel alır ve JMS ' yi gerçekleştirmek için yalnızca WebSphere MQ Version 6.0 kuyruk yöneticisinde bulunan özellikleri kullanır. Bir WebSphere MQ Sürüm 7.0 ve sonraki yayın kuyruğu yöneticilerine, WebSphere MQ ileti sistemi sağlayıcısı geçiş kipini kullanarak bağlanabilirsiniz, ancak Sürüm 7.0 eniyilemelerinin hiçbirini kullanamazsınız. Bu kip, aşağıdaki kuyruk yöneticisi sürümlerinden herhangi birine bağlantı sağlar:

1. WebSphere MQ Version 7.0, and subsequent, queue manager in bindings or client mode, but this mode uses only those features that were available to a WebSphere MQ Version 6.0 queue manager
2. İstemci kipindeki WebSphere MQ Sürüm 6.0 ya da önceki kuyruk yöneticisi

WebSphere Event Broker ya da WebSphere Message Broker ile WebSphere MQ Enterprise Transport olanağını kullanarak bağlanmak istiyorsanız, WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipini kullanın. WebSphere MQ Real-Time Transport kullanıyorsanız, bağlantı üreticisi nesnesinde belirtik olarak seçilen özelliklere sahip olduğunuz için, WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipi otomatik olarak seçilir. Connection to WebSphere Event Broker or WebSphere Message Broker using the WebSphere MQ Enterprise Transport follows the general rules for mode selection described in WebSphere MQ ileti sistemi sağlayıcısı kipini seçme kuralları .

Zamanuyumsuz ileti tüketimi

WebSphere MQ V7.0 ve sonraki yayın düzeyi zamanuyumsuz ileti tüketimini destekler. Bir uygulama, bir hedef için geri bildirme işlevini kaydedebilir. Hedefe uygun bir ileti gönderildiğinde, WebSphere MQ işlevi çağırır ve iletiyi parametre olarak iletir. Bu işlev, iletiyi zamanuyumsuz olarak işler. Önceki WebSphere MQyayınlarında, bu özellik yalnızca JMS için WebSphere MQ sınıfları kullanılırken kullanılabilir.

JMS için WebSphere MQ sınıfları, WebSphere MQ V7.0 ' da ve sonraki yayın düzeylerindeki bu yeni özelliği kötüye kullanmak için değiştirildi. JMS ileti dinleyicilerinin somutlaması artık WebSphere MQile daha doğal bir uyum sağlar ve JMS için WebSphere MQ sınıflarının artık hedefe uygun bir iletinin gönderilip gönderilmediğini denetlemek için bir hedefi yoklamak zorunda kalmadığını doğrulayın. Özellikle bir uygulama birden çok hedefi izlemek için bir oturumda birden çok ileti dinleyici kullandığında, JMS ileti dinleyicilerinin performansı bir sonuç olarak iyileştirilir. İleti verimi artırıldı ve bir hedefe ulaştıktan sonra ileti dinleyicisine bir ileti göndermek için gereken süre azaltıldı.

İletilerle yönlendirilen Bean 'ler (MDBS ' ler) benzer performans iyileştirmelerine sahiptir. Buna ek olarak, WebSphere MQ işlevine ilişkin başka bir geliştirme nedeniyle, aynı hedeften gelen iletileri tüketen birden çok MDBS ' ler ileti üzerinde çekişmeyi azaltmış olabilir.

İleti seçimi

With the exception of selecting messages by message identifier or correlation identifier, all message selection in releases of WebSphere MQ prior to Version 7.0 was done by WebSphere MQ classes for JMS. WebSphere MQ V7.0' da ve sonraki yayın düzeylerinde, tüm ileti seçimi kuyruk yöneticisi tarafından yapılır.

Sonuç olarak, ileti seçimi kullanılarak ileti tüketen uygulamalar için ileti çıkışı artırılır. Yalnızca seçim ölçütlerine uyan iletiler ağ üzerinden aktarıldığından ve JMS için WebSphere MQ sınıfları yalnızca uygulamaya teslim ettiği iletileri işleyeceğinden, performans iyileştirmesi, istemci kipinde bağlantı kuran bir uygulama için daha fazladır.

İletişim bağlantısının paylaşılması

In previous releases of WebSphere MQ, if a WebSphere MQ client application connected to a queue manager more than once using the same MQI channel, each instance of the MQI channel required a separate TCP connection. WebSphere MQ V7.0 ' da ve sonraki yayın düzeylerinde, aynı MQI kanalını kullanan kuyruk yöneticisine yapılan her bağlantı tek bir TCP bağlantısını paylaşabilir. Bu düzenleme, daha az sayıda ağ kaynağının gerekli olduğu ve kuyruk yöneticisine birden çok bağlantı yaratmak için alınan toplam sürenin, özellikle de SSL anlaşması TCP bağlantısının başlangıcındaki bir kez gerçekleştiğinden, SSL kullanılırken azaltıldığı anlamına gelir.

JMS için WebSphere MQ sınıfları bu geliştirmeyi söktürür. İstemci kipinde bir kuyruk yöneticisine bağlanan bir uygulama için, JMS için WebSphere MQ sınıfları, ConnectionFactory nesnesinin bir özelliği olarak belirtilen adı taşıyan MQI kanalını kullanarak bir kuyruk yöneticisine birden çok bağlantı yaratabilir. Kuyruk yöneticisine bu bağlantıların her biri, artık tek bir TCP bağlantısını paylaşabilirler.

İstemci bağlantılarında önden okuma

Bir uygulama, kalıcı olmayan iletileri bir hedeften tüketmek için istemci bağlantısı kullanıyorsa, hedef, JMS için WebSphere MQ sınıflarının uygulamaya teslim edilmeden önce ilgilenilecek iletileri depolamak için bir arabellek kullanmasını sağlamak üzere yapılandırılabilir. Bu eniyileme *önden okuma* olarak adlandırılır ve alma () yöntemini çağırarak ve iletileri zamanuyumsuz olarak tüketen ileti dinleyicilerine ve MDBS ' ye göre iletileri zamanuyumlu olarak tüketen uygulamalar tarafından kullanılabilir. İleriye okuma, özellikle hızla tüketilmesi gereken çok sayıda ileti içeren hedefler için etkili olur.

Önceden okuma, kalıcı iletiler için geçerli değildir; kalıcı iletiler bir arabelleğe okunduysa, kuyruk yöneticisi artık bir başarısızlığın ardından gelen iletileri kurtaramaz. Ancak, kalıcı ve kalıcı olmayan iletilerin karışımıyla bir hedeften iletileri tüketen bir uygulama, okuma öncesinde hala kullanabilir. İletilerin sırası korunur, ancak önden okuma yürütme ortamı yararları yalnızca kalıcı olmayan iletiler için geçerlidir.

Önden okuma kullanıp kullanmayacağına karar verirken, aşağıdaki noktaları göz önünde bulundurun:

- Bir uygulama, ileriye okuma için yapılandırılmış bir hedeften gelen iletileri tüketiyorsa ve uygulama herhangi bir nedenle sona ererse, arabellekte saklanmış olan kalıcı olmayan iletiler atılır.
- Aşağıdaki koşulların tümü doğru ise, bir oturumdaki kuyruğa gönderilen iletiler gönderildikleri sırayla alınmayabilir:
  - Bir uygulama, kuyruktan iletileri tüketebilmek için aynı oturumda iki ileti tüketicisini kullanır.
  - Her ileti tüketicisi, kuyruk için farklı bir Hedef nesnesi kullanır.
  - Hedef nesnelerin herhangi biri ya da her ikisi de, okuma öncesinde yapılandırılanmak üzere yapılandırılır.

İletilerin gönderilmesi

Bir uygulama bir hedefe ileti gönderdiğinde, hedef, iletiyi çağırdığında (), JMS için WebSphere MQ sınıfları kuyruk yöneticisine iletir ve kuyruk yöneticisinin iletiyi güvenli bir şekilde alıp almadığını belirlemeksizin, denetimi yeniden uygulamaya geri döndürecek şekilde yapılandırılabilir. JMS için WebSphere MQ sınıfları, yalnızca kalıcı olmayan iletiler için ve hareket edilen bir oturumda gönderilen kalıcı iletiler için bu şekilde çalışabilir.

İşlem yapılan bir oturumda gönderilen kalıcı iletiler için, uygulama, commit () çağrılırken kuyruk yöneticisinin iletileri güvenli bir şekilde alıp almadığını belirler. İşlem dışı bir oturumda gönderilen iletiler için, ConnectionFactory nesnesinin SENDCHECKCOUNT özelliği, JMS için WebSphere MQ sınıfından önce kuyruk yöneticisinin iletileri güvenli bir şekilde aldığını denetlemeden önce kaç iletinin gönderileceğini belirtir.

Bu eniyileme, istemci kipindeki bir kuyruk yöneticisine bağlanan ve bir dizi ileti dizisini hızlı bir şekilde göndermesi gereken bir uygulamaya en çok yarar sağlar, ancak gönderilen her ileti için kuyruk yöneticisinden hemen geribildirim gerektirmez.

Kanal çıkışları

JMS için WebSphere MQ sınıflarından çağrıldığında, C ya da C++ dilinde yazılmış kanal çıkış programları artık bir Websphere MQ MQI istemcisinden çağrıldığı gibi işlev görmektedir. The performance of channel exit classes written in Java has been improved, and you can now write channel exit classes using a new set of interfaces in the com.ibm.mq.exits package instead of using the interfaces in WebSphere MQ classes for Java.

İleti Özellikleri

JMS iletisi, bir üstbilgi alanları kümesinden, bir özellikler kümesinden ve uygulama verilerini içeren bir gövde ile oluşur. En az bir WebSphere MQ iletisi, bir ileti tanımlayıcısından ve uygulama verilerinden oluşur.

JMS uygulaması için bir WebSphere MQ sınıfları bir JMS iletisi gönderdiğinde, JMS için WebSphere MQ sınıfları JMS iletisi bir WebSphere MQ iletisine eşler. JMS üstbilgi alanlarının ve özelliklerinin bazıları, ileti tanımlayıcısındaki alanlarla eşlenir ve bazıları, MQRFH2 üstbilgisi olarak adlandırılan ek bir WebSphere MQ üstbilgisindeki alanlarla eşlenir. JMS uygulaması için bir WebSphere MQ sınıfları bir JMS iletisi aldığında, JMS için WebSphere MQ sınıfları ters eşlemeyi gerçekleştirir.

JMS uygulamasına ilişkin bir WebSphere MQ sınıfından ileti almak için MQI kullanan bir uygulama, bu nedenle bir MQRFH2 üstbilgisini işleyebilmelidir. If the application cannot handle an MQRFH2 header, the TARGCLIENT property of the Destination object can be set to tell WebSphere MQ classes for JMS not to include an MQRFH2 header in the WebSphere MQ messages. Ancak, MQRFH2 üstbilgisi dışlanarak, JMS üstbilgi alanlarında ve özelliklerinde tutulan bilgiler kaybedilir.

Benzer şekilde, JMS uygulamasına ilişkin bir WebSphere MQ sınıflarına ileti göndermek için MQI kullanan bir uygulamanın her iletide bir MQRFH2 üstbilgisi içermesi gerekir. Bir MQRFH2 üstbilgisi içerilmiyorsa, JMS için WebSphere MQ sınıfları yalnızca ileti tanımlayıcısındaki alanlardan türetilebilecek JMS üstbilgi alanlarını ve özelliklerini ayarlayabilir.

WebSphere MQ V7.0 , JMS uygulamaları için iletileri almak ve iletileri WebSphere MQ sınıflarına göndermek için MQI ' yı kullanan uygulamalar için ek destek sağlar.

Bir uygulama, JMS uygulaması için bir WebSphere MQ sınıflarından bir ileti almak için MQGET ' i çağırdığında, uygulama iletiyi almayı aşağıdaki yollardan birini seçebilir:

1. İleti, JMS üstbilgi alanları ve özelliklerinden türetilen verileri ve uygulama verilerini içeren bir MQRFH2 üstbilgisi olan bir ileti tanımlayıcıyla teslim edilir.
2. İleti, bir ileti tanımlayıcısı, uygulama verileri ve bir ileti özellikleri kümesi ile birlikte teslim edilir.

In option 2, each message property represents a JMS header field or property that was originally mapped by WebSphere MQ classes for JMS into a field in an MQRFH2 header. MQGET çağrısından sonra,

uygulama, ileti özelliklerinin değerlerini almak için MQINQMP çağrısını kullanabilir. Bir ileti almak için seçenek 1 yerine seçenek 2 'nin kullanılması, uygulama mantığını aşağıdaki şekillerde basitleştirir:

- Uygulamanın, JMS üstbilgi alanını ve XML benzeri bir biçimde kodlanmış özellik verilerini içeren MQRFH2 üstbilgisinin değişken bölümünü ayrıştırmak zorunda değildir.
- Uygulamanın, karakter verilerini MQRFH2 üstbilgisinin değişken kısmına dönüştürmek zorunda kalmaması gerekir.

Buna uygun olarak, bir uygulama MQPUT ' u JMS uygulaması için bir WebSphere MQ sınıflarına göndermek üzere çağırmadan önce, uygulama bir MQRFH2 üstbilgisi oluşturmak yerine ileti özellikleri değerlerini ayarlamak için MQSETMP çağrısını kullanabilir.

Hizmet verebilirlik

JMS için WebSphere MQ sınıfları, hizmet verilebilirlik ile ilgili iyileştirmeler içerir:

- İzleniyor.

  JMS için WebSphere MQ sınıfları, uygulamanın izlemeyi denetlemek için kullanabileceği bir sınıf içerir. Bir uygulama, izlemeyi başlatabilir ve durdurabilir, bir izleme sırasında gereken ayrıntı düzeyini belirleyebilir ve izleme çıkışını çeşitli yollarla uyarlayabilir.
- Günlüğe kaydetme.

  JMS için WebSphere MQ sınıflarında, düzeltmeniz gereken hatalarla ilgili iletileri içeren bir günlük dosyası vardır. İletiler düz metin olarak yazılır. JMS için WebSphere MQ sınıfları, uygulamanın günlük dosyasının konumunu ve büyüklük üst sınırını belirtmek için kullanabileceği bir sınıf içerir.
- First Failure Support Technology (FFST).

  Ciddi bir hata oluşursa, JMS için WebSphere MQ sınıfları, FDC dosyasında bir FFST raporu oluşturur. FFT raporu, IBM hizmetinin sorunu daha hızlı tanılamak için kullanabileceği bilgiler içerir.
- Sürüm bilgileri.

  JMS için WebSphere MQ sınıfları, uygulamanın JMS için WebSphere MQ sınıflarının sürümünü sorgulamak için kullanabileceği bir sınıf içerir.
- Kural dışı durum iletileri.

  Kural dışı durum iletileri, hataların nedenleriyle ve hataları düzeltmek için gereken işlemlere ilişkin daha fazla bilgi sağlamak için geliştirilmiştir.
- Uygulama sunucuları.

  The integration of the serviceability features of WebSphere MQ classes for JMS with those of WebSphere Application Server has been improved.

MQC, MQConstants ile değiştirilir

A new package, com.ibm.mq.constants, is supplied with IBM WebSphere MQ Version 7.0. Bu paket, bir dizi arabirimi gerçekleştiren MQConstants sınıfını içerir. MQConstants, MQC arabirimindeki tüm değişmezlerin ve yeni değişmezlerin tanımlamalarını içerir. Bu paketteki arabirimler, IBM WebSphere MQ' ta kullanılan sabitler üstbilgi dosyalarının adlarını yakından takip eder.

Örneğin, CMQC arabirimi sabit bir MQOO_INPUT_SHARED; bu arabirimi ve değişmezi `cmqc.h` üstbilgi kütüğüne ve MQOO_INPUT_SHARED değişmezine karşılık gelir.

com.ibm.mq.constants can be used with both IBM WebSphere MQ classes for Java and IBM WebSphere MQ classes for JMS.

MQC hala var ve daha önce sahip olduğu sabitleri içeriyor. Ancak, yeni uygulamalar için, com.ibm.mq.constants paketini kullanmanız gerekir.

JMS uygulamaları için WebSphere MQ sınıflarının yazılması

JMS modeline kısa bir giriş yaptıktan sonra bu konu, JMS uygulamaları için WebSphere MQ sınıflarının nasıl yazılacağı hakkında ayrıntılı kılavuz sağlar.

JMS modeli

JMS modeli, Java uygulamalarının ileti alışverişi işlemlerini gerçekleştirmek için kullanabileceği bir arabirim kümesi tanımlar. JMS sağlayıcısı olarak JMS için WebSphere MQ sınıfları, JMS nesnelerinin WebSphere MQ kavramlarıyla nasıl ilgili olduğunu tanımlar. JMS belirtimi, belirli JMS nesnelerinin yönetilecek nesneler olmasını bekler.

JMS belirtimi ve javax.jms paketi, Java uygulamalarının ileti alışverişi işlemlerini gerçekleştirmek için kullanabilecekleri bir arabirim kümesi tanımlar. Aşağıdaki liste ana JMS arabirimlerini özetlemektedir:

Hedef
Bir hedef, bir uygulamanın iletileri gönderdiği ya da bir uygulamanın iletileri aldığı bir kaynaktır ya da her ikisi de olabilir.

ConnectionFactory
Bir ConnectionFactory nesnesi, bir bağlantı için bir yapılandırma özellikleri kümesini sarsalıyor. Bir uygulama, bağlantı oluşturmak için bir bağlantı üreticisi kullanır.

Bağlantı
Bir bağlantı nesnesi, bir uygulamanın ileti sistemi ile etkin bağlantısını sarsalıyor. Uygulama, oturum yaratmak için bir bağlantı kullanır.

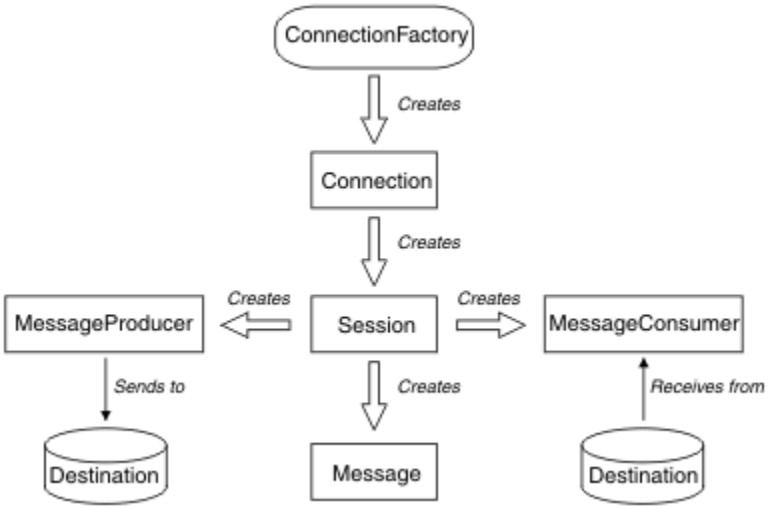
Oturum
Oturum, ileti göndermek ve almak için tek bir iş parçacıklı bağlamdır. Bir uygulama, ileti, ileti üreticileri ve ileti tüketicileri oluşturmak için bir oturumu kullanır. Bir oturum hareket edilir ya da hareket edilmez.

İleti
Bir ileti nesnesi, bir uygulamanın gönderdiği ya da gönderdiği bir iletiyi sarsalıyor.

MessageProducer
Bir uygulama, iletileri bir hedefe göndermek için bir ileti üreticiyi kullanır.

MessageConsumer
Bir uygulama, bir hedefe gönderilen iletileri almak için bir ileti tüketicisi kullanır.

Şekil 127 sayfa 773 , bu nesneleri ve bunların ilişkilerini gösterir.



Şekil 127. JMS nesneleri ve ilişkileri

Bir Hedef, ConnectionFactoryya da Connection nesnesi, çok iş parçacıklı bir uygulamanın farklı iş parçacıkları tarafından eşzamanlı olarak kullanılabilir, ancak bir Oturum, MessageProducerya da

MessageConsumer nesnesi farklı iş parçacıkları tarafından koşutzamanlı olarak kullanılamaz. Bir Oturum, MessageProducerya da MessageConsumer nesnesinin koşutzamanlı olarak kullanılmamasını sağlamanın en basit yolu, her iş parçacığı için ayrı bir Oturum nesnesi yaratmamaktadır.

JMS, iki ileti alışverişi biçemlerini destekler:

- Noktadan Noktaya İleti Sistemi
- Yayınlama/abone olma ileti alışverişi

Bu ileti alışverişi biçemleri de *ileti sistemi etki alanları*olarak da adlandırılır ve her iki ileti sistemi stilini de bir uygulamada birleştirebilirsiniz. Noktadan noktaya iletişim alanında, hedef bir kuyruktır ve yayınlama/abone olma etki alanında, hedef bir konudur.

JMS 1.1öncesi JMS sürümleriyle, noktadan noktaya iletişim etki alanı için programlama bir arabirim ve yöntem kümesini kullanır ve yayınlama/abone olma etki alanı için programlama başka bir küme kullanır. İki set birbirine benzer, ama ayrı. JMS 1.1ile, hem ileti sistemi etki alanlarını destekleyen, ortak bir arabirim ve yöntem kümesi kullanabilirsiniz. Ortak arabirimler, her ileti sistemi etki alanı için bir etki alanı bağımsız görünümü sağlar. Çizelge 103 sayfa 774 , JMS etki alanı bağımsız arabirimlerini ve bunlara karşılık gelen etki alanlarını özel arabirimlerini listeler.

Çizelge 103. JMS etki alanı bağımsız ve etki alanına özgü arabirimler

Etki alanı bağımsız arabirimleri	**Noktadan noktaya etki alanına ilişkin etki alanına özgü arabirimler**	**Yayınlama/abone olma etki alanı için etki alanına özgü arabirimler**
ConnectionFactory	QueueConnectionÜreticisi	TopicConnectionÜreticisi
Bağlantı	QueueConnection	TopicConnection
Hedef	Kuyruk	Konu
Oturum	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver QueueBrowser	TopicSubscriber

JMS 1.1 , etki alanına özgü tüm arabirimleri korur ve bu nedenle var olan uygulamalar bu arabirimleri kullanmaya devam edebilir. Ancak yeni uygulamalar için, etki alanı bağımsız arabirimlerini kullanmayı düşünün.

JMS için WebSphere MQ sınıflarında, JMS nesneleri WebSphere MQ kavramlarıyla aşağıdaki şekillerde ilişkilendirilir:

- Bağlantı nesnesinin, bağlantıyı yaratmak için kullanılan bağlantı üreticisinin özelliklerinden türetilmiş özellikleri vardır. Bu özellikler, bir uygulamanın kuyruk yöneticisine nasıl bağlanacağını denetler. Bu özelliklere örnek olarak, kuyruk yöneticisinin adı ve istemci kipinde kuyruk yöneticisine bağlanan bir uygulama için, kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.
- Bir Oturum nesnesi, oturumun işlemsel kapsamını tanımlayan bir WebSphere MQ bağlantı tanıtıcısını sarsacaktır.
- Her biri bir WebSphere MQ nesne tanıtıcısını sarmalayan bir MessageProducer nesnesi ve bir MessageConsumer nesnesi.

JMS için WebSphere MQ sınıflarını kullanırken, WebSphere MQ ' un normal tüm kuralları geçerli olur. Özellikle, bir uygulamanın uzak bir kuyruğa ileti gönderebileceği, ancak yalnızca, uygulamanın bağlı olduğu kuyruk yöneticisinin sahip olduğu bir kuyruktan bir ileti alabileceği unutulmamalı.

JMS belirtimi, ConnectionFactory ve Hedef nesnelerin denetlenmesini bekler. Bir denetimci, denetlenen nesneleri merkezi bir havuzda yaratır ve bakımını yapar; JMS uygulaması bu nesneleri Java Naming and Directory Interface (JNDI) kullanarak alır.

JMS için WebSphere MQ sınıflarında, Hedef arabirimin somutlaması, Kuyruk ve Konu 'nın soyut bir üst sınıfıdır ve bu nedenle, Hedef eşgörünümü bir Kuyruk nesnesi ya da Konu nesnesidir. Etki alanı bağımsız arabirimleri, bir kuyruğu ya da konuyu hedef olarak kabul eder. Bir MessageProducer ya da MessageConsumer nesnesine ilişkin ileti alışverişi etki alanı, hedefin bir kuyruk mu, yoksa bir konu mı tarafından belirlenir.

In WebSphere MQ classes for JMS therefore, objects of the following types can be administered objects:

- ConnectionFactory
- QueueConnectionÜreticisi
- TopicConnectionÜreticisi
- Kuyruk
- Konu
- XAConnectionFactory
- XAQueueConnectionÜreticisi
- XATopicConnectionÜreticisi

JMS iletileri

JMS iletileri bir üstbilgiden, özelliklerden ve bir gövden oluşur. JMS, ileti gövdesinin beş tipini tanımlar.

JMS iletileri aşağıdaki kısımlardan oluşur:

Üstbilgi
Tüm iletiler aynı üstbilgi alanları kümesini destekler. Üstbilgi alanları, iletileri tanımlamak ve yönlendirmek için hem istemciler, hem de sağlayıcılar tarafından kullanılan değerleri içerir.

Özellikler
Her ileti, uygulama tanımlı özellik değerlerini desteklemek için yerleşik bir olanak içerir. Özellikler, uygulama tanımlı iletilere süzgeç uygulamak için verimli bir mekanizma sağlar.

Gövde
JMS, şu anda kullanılmakta olan ileti sistemi stillerinin çoğunu kapsayan çeşitli ileti gövdesi tiplerini tanımlar.

JMS, beş ileti gövdesi tipini tanımlar:

Akış
Java temel değerlerinin akışıdır. Doldurulan ve sıralı bir şekilde okunuyor.

Eşlem
Adların dizgi ve değer olduğu ad-değer çiftleri kümesi, Java temel tipleridir. Girişlere sırayla ya da rasgele ad temelinde erişilebilir. Girişlerin sırası tanımsız.

Metin
A message containing a java.lang.String.

Nesne
Diziselleşebilir bir Java nesnesi içeren bir ileti

Bayt
Yorumlanmamış byte akışıdır. Bu ileti tipi, bir gövenin var olan bir ileti biçimiyle eşleşmesi için tam olarak kodlanır.

JMSCorrelationID üstbilgi alanı, bir iletiyi başka bir iletiyi bağlamak için kullanılır. Bu ileti genellikle, bir yanıt iletisini istek isteyen iletisiyle bağlantıdır. JMSCorrelationID , sağlayıcıya özgü bir ileti tanıtıcısını, uygulamaya özgü bir dizgiyi ya da bir sağlayıcı-yerel bayt [] değerini barınabiliyor.

JMS ' de ileti seçicileri

İletiler, uygulama tanımlı özellik değerleri içerebilir. Bir uygulama, JMS sağlayıcısı süzgeç iletilerine sahip olmak için ileti seçicileri kullanabilir.

İleti, uygulama tanımlı özellik değerlerini desteklemek için yerleşik bir olanak içerir. Sonuç olarak, bu, uygulamaya özgü üstbilgi alanlarını bir iletiye eklemek için bir mekanizma sağlar. Özellikler, ileti

seçicilerini kullanarak bir JMS sağlayıcısının kendi adına iletileri seçmesine ya da uygulamaya özgü ölçütlere göre süzgeç uygulamasına olanak tanır. Uygulama tanımlı özellikler aşağıdaki kurallara uymalıdır:

- Özellik adları, bir ileti seçici tanıtıcısına ilişkin kurallara uymalıdır.
- Özellik değerleri Boole, bayt, short, int, long, float, double, ve String olabilir.
- JMSX ve JMS_ ad önekleri ayrılmıştır.

Özellik değerleri, bir ileti göndermeden önce ayarlanır. Bir istemci bir ileti aldığında, ileti özellikleri salt okunurdur. Bir istemci bu noktada özellikleri ayarlama girişiminde bulunursa, bir MessageNotWriteableException yayınlanır. clearProperties çağrılırsa, özellikler artık hem okunabilecek, hem de yazılacak şekilde yazılabilir.

Bir özellik değeri, ileti gövdesindeki bir değeri yineleyebilir. JMS, bir özelliğe yapılmanın yapılabileceği bir ilke tanımlamıyor. Ancak, uygulama geliştiricilerin, JMS sağlayıcılarının bir ileti gövdesindeki verileri ileti özelliklerinde verilerden daha verimli bir şekilde işleyeceğini dikkate almaları gerekir. En iyi performans için, uygulamaların ileti özelliklerini yalnızca bir ileti üstbilgisini özelleştirmeleri gerektiğinde kullanması gerekir. Bunu yapmanın birincil nedeni, uyarlanmış ileti seçimini desteklemesidir.

JMS ileti seçici, bir istemcinin ileti üstbilgisini kullanarak, ilgilendiği iletileri belirtmesine olanak sağlar. Yalnızca, seçiciyle eşleşen üstbilgileri içeren iletiler teslim edilir.

İleti seçicileri, ileti gövdesi değerlerine gönderme yapamazlar.

Bir ileti seçici, ileti üstbilgisi alanı ve özellik değerleri, seçicide karşılık gelen tanıtıcılarının yerine geçtiğinde, seçici doğru olarak değerlendirildiğinde bir iletiyle eşleşir.

İleti seçici, sözdizimi, SQL92 koşullu ifade sözdiziminin bir alt kümesine dayalı olan bir Dizedir. Bir ileti seçicinin değerlendirdiği sıra, öncelik düzeyi içinde soldan sağa doğru olur. Bu siparişi değiştirmek için ayraçları kullanabilirsiniz. Önceden tanımlı seçici hazır bilgileri ve işleç adları burada büyük harfle yazılır; ancak, büyük/küçük harf duyarlı değildir.

Bir seçici şunları içerebilir:

- Hazır Bilgiler
  - Dizgi hazır bilgisi tırnak içine alınır. Çift tırnak işareti, bir tırnak işaretini temsil eder. 'Literal' ve 'literal' (hazır bilgi) örnekleri verilebilir. Java dizgi hazır bilgileri gibi, bunlar Unicode karakter kodlamasını kullanır.
  - Tam sayısal hazır bilgi, 57, -957 ve +62 gibi ondalık bir nokta içermeyen bir sayısal değerdir. Java uzun aralıklarındaki sayılar desteklenir.
  - Yaklaşık sayısal hazır bilgi, bilimsel gösterimde (7E3 ya da -57.9E2gibi) sayısal bir değerdir ya da 7., -95.7ya da +6.2gibi bir ondalık değer. Java double (Çift) aralığındaki numaralar desteklenir.
  - Boole tipi hazır bilgiler TRUE ve FALSE değerini verir.
- Tanıtıcılar:
  - Tanıtıcı, Java harflerinden biri olması gereken Java harflerinin ve Java sayılarının sınırsız uzunluktaki sıralarından biri. Bir harf, Character.isJavaLetter yönteminin true (doğru) değerini döndürdüğü herhangi bir karakterdir. Bu, _ ve $' ı içerir. Bir harf ya da rakam, Character.isJavaLetterOrDigit yönteminin true (doğru) değerini döndürdüğü herhangi bir karakterdir.
  - Tanıtıcılar NULL, TRUE ya da FALSE adları olamaz.
  - Tanıtıcılar NOT, AND, OR, BETWE, LIKE, IN ya da IS olamaz.
  - Tanıtıcılar, üstbilgi alanı başvuruları ya da özellik başvurularıdır.
  - Tanıtıcılar büyük ve küçük harfe duyarlıdır.
  - İleti üstbilgisi alanı başvuruları şu şekilde kısıtlanmıştır:
    - JMSDeliveryMode
    - JMSönceliği
    - JMSMessageID
    - JMSTimestamp

- JMSCorrelationID
- JMSType

JMSMessageID, JMSTimestamp, JMSCorrelationIDve JMSType değerleri boş değerli olabilir ve bu durumda boş değer (NULL) olarak işlem görür.

  - JMSX ile başlayan herhangi bir ad, JMS tanımlı bir özellik adıdır.
  - JMS_ ile başlayan herhangi bir ad, sağlayıcıya özgü bir özellik adıdır.
  - JMS ile başlamayan herhangi bir ad, uygulamaya özgü bir özellik adıdır. Bir iletide var olmayan bir özelliğe ilişkin başvuru varsa, değeri NULL (boş değerli) olur. Böyle bir değer varsa, bu değer, ilgili özellik değeridir.
- Beyaz alan, Java için tanımlanla aynıdır: boşluk, yatay sekme, form besleme ve satır sonlandırıcı.
- İfadeler:
  - Seçici, koşullu bir ifadedir. Gerçek eşleşmeleri değerlendiren bir seçici; yanlış ya da bilinmeyen olarak değerlendirilen bir seçici eşleşmez.
  - Aritmetik ifadeler, kendilerinden, aritmetik işlemlerden, tanıtıcılardan (sayısal hazır bilgi olarak kabul edilen bir değere sahip) ve sayısal hazır bilgiler içerir.
  - Koşullu ifadeler kendilerinden, karşılaştırma işlemlerinden ve mantıksal işlemlerden oluşur.
- İfadelerin değerlendirilen sıralamayı ayarlamak için standart destek pazarlama () desteklenir.
- Mantıksal işleçler öncelik sırasına göre: NOT, AND, OR.
- Karşılaştırma işleçleri: =, >, > =, <, < =, < > (eşit değil).
  - Yalnızca aynı tipteki değerler karşılaştırılabilir. Bir kural dışı durum, tam sayısal değerleri ve yaklaşık sayısal değerleri karşılaştırmak için geçerli bir değerdir. (Gereken tip dönüşümü, Java sayısal promosyonunun kuralları tarafından tanımlanır.) Farklı tipleri karşılaştırma girişiminde bulunulursa, seçici her zaman false olur.
  - Dizgi ve Boole karşılaştırması = ve < > ile kısıtlanmıştır. İki dizgi, yalnızca aynı karakter dizisini içeriyorsa eşittir.
- Aritmetik işleçler öncelik sırasına göre:
  - +,-birli.
  - *,/, çarpma ve bölme.
  - +,-, toplama ve çıkarma.
  - Boş değer (NULL) olan aritmetik işlemler desteklenmez. Bunlar denenirse, tam seçici her zaman false olur.
  - Aritmetik işlemler, Java sayısal promosyonu kullanmalıdır.
- arithmetic-expr1 [NOT] BETWORN arithmetic-expr2 ve arithmetic-expr3 karşılaştırma işleci:
  - 15-19 yaş arası, yaş > = 15 ve yaş < = 19 olarak eşittir.
  - 15-19 yaş değerleri, < 15 YA DA age > 19 yaşının eşdeğeridir.
  - BETWARY işleminin ifadelerinden herhangi biri NULL (boş) ise, işlemin değeri false olur. BETWAY işlemi NOT NULL ifadelerinden herhangi biri NULL (boş değer) ise, işlemin değeri true olur.
- tanıtıcı [NOT] IN (string-literal1, string-literal2, ...) tanıtıcıyı bir Dize ya da NULL değerine sahip olan karşılaştırma işleci.
  - Country IN ('UK ',' US ',' France '),' UK için 've' Peru 'için' yanlış ' için geçerlidir. Bu ifade (Country = 'UK') OR (Country = 'US ') YA DA (Country = 'France') ifadesine eşdeğerdir.
  - Country NOT IN ('UK ',' US ',' France ')' UK 'için yanlış ve' Peru için doğru '. NOT ((Ülke = 'UK ') YA DA (Ülke = 'ABD') YA DA (Ülke = 'Fransa ')) ifadelerine eşdeğerdir.
  - IN ya da NOT IN işlemi tanıtıcısı NULL ise, işlemin değeri bilinmez.
- Tanıtıcı [NOT] LIKE örüntü-değeri [ESCAPE escape-charactery] karşılaştırma işleci, burada tanıtıcının dizgi değeri var. pattern-value bir dizgi hazır bilgisidir; burada _ herhangi bir tek karakter için ve% herhangi bir karakter dizisi anlamına gelir (boş sıra dahil). Diğer tüm karakterler kendileri için geçerli.

İsteğe bağlı çıkış karakteri tek bir karakter dizgisi hazır bilgisidir; örüntüdeki _ ve% karakterlerinin özel anlamlarından kaçmak için kullanılan karakter.

  - '12%3' gibi bir telefon 123 ve 12993 için geçerlidir ve 1234 için false değerini içerir.
  - 'l_se' gibi sözcük, "kaybetmek" için doğrudur ve "gevşek" için "false".
  - '_ %' ESCAPE' \ ', "_foo" için "true" ve "bar" için false değerini içerir.
  - '12%3', 123 ve 12993 için yanlış ve 1234 için doğru (true) olarak değil.
  - LIKE ya da NOT LIKE işleminin tanıtıcısı NULL ise, işlemin değeri bilinmiyor demektir.
- tanıtıcı, boş değerli bir üstbilgi alanı değeri ya da eksik bir özellik değeri için boş değerli (NULL) karşılaştırma işleci sınamaları.
  - prop_name IS NULL.
- tanıtıcı, boş değerli olmayan bir üstbilgi alan değeri ya da bir özellik değeri varlığı için NOT NULL karşılaştırma işleci sınamaları DEĞİL.
  - prop_name BOŞ DEĞİL.

Aşağıdaki ileti seçici, ileti tipi araba, mavi renk ve ağırlığı 2500 lbs 'den büyük olan iletileri seçer:

```
"JMSType = 'car' AND color = 'blue' AND weight > 2500"
```

Yukarıdaki listede belirtildiği gibi, özellik değerleri NULL (boş değerli) olabilir. NULL değerler içeren seçici ifadelerinin değerlendirilmesi, SQL 92 NULL anlambilimi ile tanımlanır. Aşağıdaki liste, bu anlambilimi hakkında kısa bir açıklama sağlar:

- SQL, bir NULL değerini bilinmeyen olarak değerlendirir.
- Bilinmeyen bir değere sahip karşılaştırma ya da aritmetik, her zaman bilinmeyen bir değer verir.
- IS NULL işleci, bilinmeyen bir değeri TRUE değerine dönüştürür.
- IS NOT NULL işleci, bilinmeyen bir değeri FALSE değerine dönüştürür.

SQL, sabit ondalık karşılaştırma ve aritmetik desteklerini desteklese de, JMS ileti seçicileri desteklemez. Bu nedenle, tam sayısal hazır bilgiler ondalıklı olmayan bu sayılarla sınırlandırılmıştır. Ayrıca, yaklaşık bir sayısal değer için alternatif bir temsil olarak bir ondalık ile sayısal değerler olmasının nedeni de bu.

SQL açıklamaları desteklenmez.

JMS iletilerinin WebSphere MQ iletilerine eşlenmesi

WebSphere MQ iletileri, bir ileti tanımlayıcısı, isteğe bağlı bir MQRFH2 üstbilgisinden ve bir gövdesinden oluşur. Bir JMS iletisinin içeriği kısmen eşlenmiş ve bir WebSphere MQ iletisine kısmen kopyalanmıştır.

Bu konuda, bu bölümün ilk kısmında açıklanan JMS iletisi yapısının bir WebSphere MQ iletisine nasıl eşlendiği ele alınmıştır. JMS ve geleneksel WebSphere MQ uygulamaları arasında ileti göndermek isteyen programcıların ilgisini çekmektedir. Ayrıca, iki JMS uygulaması arasında iletilen iletileri (örneğin, bir WebSphere Message Broker somutlaması) işlemek isteyen kişilerin de ilgisini çekmektedir.

Bir uygulama bir aracıya gerçek zamanlı bağlantı kullanıyorsa bu bölüm geçerli değildir. Bir uygulama gerçek zamanlı bir bağlantı kullandığında, tüm iletişim doğrudan TCP/IP; üzerinden gerçekleştirilir; WebSphere MQ kuyrukları ya da iletileri bu işe karışmaz.

WebSphere MQ iletileri üç bileşenden oluşur:

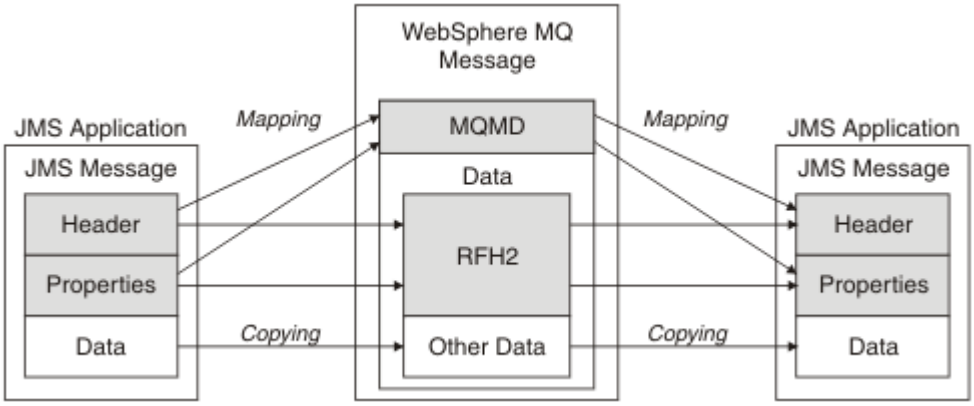
- WebSphere MQ Message Descriptor (MQMD)
- Bir WebSphere MQ MQRFH2 üstbilgisi
- Mesaj gövdesi.

MQRFH2 isteğe bağlıdır ve giden bir iletiye dahil edilmesi, JMS Hedef sınıfındaki bir işaretle yönetilir. Bu işareti, WebSphere MQ JMS yönetim aracını kullanarak ayarlayabilirsiniz. MQRFH2 , JMS ' ye özgü bilgileri taşıdığından, gönderen hedefin bir JMS uygulaması olduğunu gönderdiğinde her zaman iletiyi iletiye dahil

edin. Olağan durumda, bir iletiyi doğrudan JMS dışı bir uygulamaya gönderirken MQRFH2 ' yi atlayın. Bunun nedeni, böyle bir uygulamanın WebSphere MQ iletisinde bir MQRFH2 beklememesi olabilir.

Gelen bir iletinin MQRFH2 üstbilgisi yoksa, iletinin JMSReplyTo üstbilgisinden türetilmiş olan Kuyruk ya da Konu nesnesi varsayılan olarak bu işaret kümesine sahiptir; böylece, kuyruğa ya da konuya gönderilen bir yanıt iletisinin bir MQRFH2 üstbilgisi yoktur. Bir yanıt iletisinde MQRFH2 üstbilgisi dahil olmak üzere bu davranışı, özgün iletinin bir MQRFH2 üstbilgisi varsa, bağlantı üreticisinin TARGCLIENTMATCHESLE özelliğini NO olarak ayarlayarak değiştirebilirsiniz.

Şekil 128 sayfa 779 , JMS iletisinin yapısının bir WebSphere MQ iletisine nasıl dönüştürülecağını ve yeniden nasıl dönüştürülecağını gösterir:



Şekil 128. İletiler, MQRFH2 üstbilgisini kullanarak JMS ile WebSphere MQ arasında nasıl dönüştürülebiliyor

Yapılar iki şekilde dönüştürülebiliyor:

Eşleme
MQMD ' nin JMS alanıyla eşdeğeri olan bir alanı içerdiği durumlarda, JMS alanı MQMD alanına eşlenir. JMS uygulaması, JMS dışı bir uygulamayla iletişim kurarken bu alanları ayarlamaya ya da ayarlamaya gerek dubileceğinden, ek MQMD alanları JMS özellikleri olarak gösterilir.

Kopyalama
Where there is no MQMD equivalent, a JMS header field or property is passed, possibly transformed, as a field inside the MQRFH2.

MQRFH2 üstbilgisi ve JMS
Bu konu grubunda, ileti içeriğiyle ilişkili JMS ' ye özgü verileri taşıyan MQRFH Sürüm 2 üstbilgisi açıklanır. MQRFH2 Sürüm 2, genişletilebilir bir üstbilgidir ve JMS ile doğrudan ilişkilendirilmemiş ek bilgileri de taşıyabilir. Ancak bu bölüm yalnızca JMS ' ye göre kullanımını kapsar.Tam tanım için bkz. MQRFH2 -Kural ve biçimleme üstbilgisi 2.

Üstbilginin iki bölümü, sabit bir kısmı ve değişken kısmı vardır.

Sabit bölüm
Sabit bölüm, *standart* WebSphere MQ üstbilgi örüntüleriyle modellenir ve aşağıdaki alanlardan oluşur:

StrucId (MQCHAR4)
Yapı tanıtıcısı.

MQRFH_STRUC_ID değeri olmalıdır (değer: "RFH ") (başlangıç değeri).

MQRFH_STRUC_ID_ARRAY (değer: "R","F","H"," ") de tanımlı.

Sürüm (MQUZE)
Yapı sürüm numarası.

MQRFH_VERSION_2 (değer: 2) (başlangıç değeri) olmalıdır.

StrucLength (MQUZE)
NameValueVeri alanları da içinde olmak üzere toplam MQRFH2uzunluğu.

StrucLength olarak ayarlanan değer, birden çok 4 olmalıdır (bunu başarmak için NameValueVeri alanlarındaki veriler boşluk karakterleriyle doldurulabilir).

Kodlama (MQUZE)
Veri kodlaması.

Encoding of any numeric data in the portion of the message following the MQRFH2 (the next header, or the message data following this header).

CodedCharSetId (MQHOMER)
Kodlanmış karakter takımı tanıtıcısı.

Representation of any character data in the portion of the message following the MQRFH2 (the next header, or the message data following this header).

Biçim (MQCHAR8)
Biçim adı.

Format name for the portion of the message following the MQRFH2.

İşaretler (MQUZE)
Bayraklar.

MQRFH_NO_FLAGS = 0. İşaret ayarlanmadı.

NameValueCCSID (MQUZE)
Bu üstbilgide bulunan NameValueVeri karakteri dizgilerine ilişkin kodlanmış karakter takımı tanıtıcısı (CCSID). NameValueVerileri, üstbilgide bulunan diğer karakter dizgilerinden farklı olan bir karakter kümesinde kodlanabilir (StrucID ve Format).

NameValueCCSID 'si 2 baytlık bir Unicode CCSID 'se (1200, 13488 ya da 17584), Unicode 'un bayt sırası MQRFH2içindeki sayısal alanların bayt sıralamasını aynıdır. (Örneğin, Sürüm, StrucLengthve NameValueCCSID ' nin kendisi.)

Çizelge 104. NameValueCCSID alanı için olası değerler

Değer	Anlamı
1200	UCS2 açık uçlu
1208	UTF8
13488	UCS2 2.0 altkümesi
17584	UCS2 2.1 altkümesi (Euro simgesini içerir)

Değişken kısmı
Değişken kısmı sabit porsiyondan sonra gelir. Değişken kısmı, değişken sayıda MQRFH2 klasörü içeriyor. Her klasör, bir değişken öğe sayısı ya da özellik içerir. Klasörler grubuna ilişkin özellikler. JMS tarafından yaratılan MQRFH2 üstbilgileri aşağıdaki klasörlerden herhangi birini içerebilir:

< mcd> klasörü

`mcd` , iletinin biçimini açıklayan özellikleri içerir. Örneğin, ileti hizmeti etki alanı `Msd` özelliği bir JMS iletisini `JMSTextMessage`, `JMSBytesMessage`, `JMSStreamMessage`, `JMSMapMessage`, `JMSObjectMessage`eya da boş değerli olarak tanımlar.

`mcd` klasörü her zaman, `MQRFH2`içeren bir JMS iletisinde bulunur.

Bu, her zaman WebSphere Message Broker 'dan gönderilen bir `MQRFH2` iletisi içeren bir iletidir. Bu belge, bir iletinin etki alanını, biçimini, tipini ve ileti kümesini açıklar.

Çizelge 105. mcd özellik adı, eşanlamlı, veri tipi ve klasör			
Özellik eşanlamlı	**Özellik adı**	**Veri tipi**	**Klasör**
	mcd.Msd	string	<mcd><Msd>*messageDomain*</Msd></mcd>
	mcd.Set	string	<mcd><Set>*messageDomain*</Set></mcd>
	mcd.Type	string	<mcd><Type>*messageDomain*</Type></mcd>
	mcd.Fmt	string	<mcd><Fmt>*messageDomain*</Fmt></mcd>

mcd klasörüne kendi özelliklerinizi eklemeyin.

< jms> klasörü

jms , JMS üstbilgi alanlarını ve MQMD' da tam olarak ifade edilemeyen JMSX özelliklerini içerir. jms klasörü her zaman bir JMS MQRFH2içinde bulunur.

< usr> klasörü

usr , iletiyle ilişkili uygulama tanımlı JMS özelliklerini içerir. usr klasörü yalnızca, uygulama için uygulama tanımlı bir özellik ayarlandıysa bulunur.

< mqext> klasörü

mqext , yalnızca WebSphere Application Server tarafından kullanılan özellikleri içerir. Bu klasör yalnızca, uygulama IBM tanımlı özelliklerden en az birini ayarladıysa var olur.

Çizelge 106. mqext özellik adı, eşanlamlı, veri tipi ve klasör			
Özellik eşanlamlı	**Özellik adı**	**Veri tipi**	**Klasör**
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>*wrmCorrelator*</Wrm></mqext>

mqext klasörüne kendi özelliklerinizi eklemeyin.

< mqps> klasörü

mqps , yalnızca IBM WebSphere MQ yayınlama/abone olma yoluyla kullanılan özellikleri içerir. Bu klasör, yalnızca uygulama tümleşik yayınlama/abone olma özelliklerinden en az birini ayarladıysa var olur.

Çizelge 107. mqps özellik adı, eşanlamlı, veri tipi ve klasör			
Özellik eşanlamlı	**Özellik adı**	**Veri tipi**	**Klasör**
MQTopicString	mqps.Top	string	<mqps><Top>*topicString*</Top></mqps>
MQSubUserVerileri	mqps.Sud	string	<mqps><Sud>*subscriberUserData*</Sud></mqps>
MQIsRetained	mqps.Ret	boolean	<mqps><Ret>*isRetained*</Ret></mqps>

Çizelge 107. mqps özellik adı, eşanlamlı, veri tipi ve klasör (devamı var)			
Özellik eşanlamlı	**Özellik adı**	**Veri tipi**	**Klasör**
MQPubOptions	mqps.Pub	i8	<mqps><Pub>*publicationOptions*</Pub></mqps>
MQPubLevel	mqps.Pbl	i8	<mqps><Pbl>*publicationLevel*</Pbl></mqps>
MQPubTime	mqpse.Pts	string	<mqps><Pts>*publicationTime*</Pts></mqps>
MQPubSeqNum	mqpse.Seq	i8	<mqps><Seq>*publicationSequenceNumber*</Seq></mqps>
MQPubStrIntData	mqpse.Sid	string	<mqps><Sid>*publicationData*</Sid></mqps>
MQPubFormat	mqpse.Pfmt	i8	<mqps><Pfmt>*messageFormat*</Pfmt></mqps>

mqps klasörüne kendi özelliklerinizi eklemeyin.

Çizelge 108 sayfa 782 , özellik adlarının tam listesini gösterir.

Çizelge 108. JMS tarafından kullanılanMQRFH2 klasörleri ve özellikleri				
JMS alan adı	**Java tipi**	**MQRFH2 klasör adı**	**Özellik adı**	**Tip/değerler**
JMSHedef	Hedef	JMS	Dst	dizgi
JMSSüresi	uzun	JMS	ÜS	i8
JMSönceliği	int	JMS	Pri	i4
JMSDeliveryMode	int	JMS	Dlv	i4
JMSCorrelationID	Dizgi	JMS	Cid	dizgi
JMSReplyTo	Hedef	JMS	Rto	dizgi
JMSTimestamp	uzun	JMS	Tms	i8
JMSType	Dizgi	MCD	Tip, Küme, Fmt	dizgi
JMSXGroupID	Dizgi	JMS	GID	dizgi
JMSXGroupSeq	int	JMS	Seq	i4
xxx (kullanıcı tanımlı)	Herhangi	USR	xxx	Herhangi Biri
		MCD	MSD	jms_yok jms_metin jms_byte jms_map jms_akımı jms_nesnesi

NameValueUzunluk (MQUZE)
Bu uzunluk alanını hemen izleyen NameValueveri dizgisinin bayt cinsinden uzunluğu (kendi uzunluğunu kapsamaz).

NameValueVerileri (MQCHARn)

Önceki NameValueUzunluk alanı tarafından bayt cinsinden uzunluğu verilen tek bir karakter dizgisi. Bir özellik dizisini tutan bir klasör içerir. Her özellik, adı klasör adı olan bir XML öğesinin içinde yer alan bir ad/tip/değer üçlüdür:

```
 <foldername>
triplet1 triplet2 .....   tripletn </foldername>
```

Kapanış `</foldername>` etiketi, dolgu karakteri olarak boşluklarla izlenebilir. Her bir kırpma, XML benzeri bir sözdizimi kullanılarak kodlanır:

```
   <name dt='datatype'>value</name>
```

Veri tipi önceden tanımlı olduğundan, `dt='datatype'` öğesi isteğe bağlıdır ve birçok özellik için atlanır. İçerilirse, `dt=` etiketinden önce bir ya da daha çok boşluk karakteri eklenmelidir.

name
Özelliğin adı; bkz. Çizelge 108 sayfa 782.

datatype
katlama işleminden sonra, Çizelge 109 sayfa 783içinde listelenen veri tiplerinden biri eşleşmelidir.

value
Çizelge 109 sayfa 783içindeki tanımlamaları kullanarak, aktarılacak değerin bir dizgi gösterimidir.

Boş değer, aşağıdaki sözdizimi kullanılarak kodlanır:

```
<name dt='datatype' xsi:nil='true'></name>
```

`xsi:nil='false'`kullanmayın.

Çizelge 109. Özellik veri tipleri

Veri tipi	Tanımlama
dizgi	< ve &dışında herhangi bir karakter dizisi
boole	`0` ya da 1 karakteri (`0` = false, 1 = true)
bin.hex	Sekizlileri temsil eden onaltılık basamaklar
i1	A number, expressed using digits `0..`9, with optional sign (no fractions or exponent). Dahil -128-127 aralığında (bu değerler de içinde olmak üzere) olmalıdır
i2	A number, expressed using digits `0..`9, with optional sign (no fractions or exponent). -32768 ile 32767 arasında (bu değerler de içinde olmak üzere) yalan olmalıdır
i4	A number, expressed using digits `0..`9, with optional sign (no fractions or exponent). -2147483648 ile 2147483647 (dahil) arasında yalan olmalıdır
i8	A number, expressed using digits `0..`9, with optional sign (no fractions or exponent). -9223372036854775808 ile 9223372036854775807 (dahil) aralığında yatmalı
int	A number, expressed using digits `0..`9, with optional sign (no fractions or exponent). i8ile aynı aralıkta olmalıdır. Gönderenin belirli bir duyarlığı özellik ile ilişkilendirmek istemiyorsa, i * tiplerinden birinin yerine kullanılabilir
r4	Kayar noktalı sayı, büyüklük < = 3.40282347E+38, > = 1.175E-37 , `0..`9basamakları kullanılarak ifade edilir, isteğe bağlı işaret, isteğe bağlı kesirli basamaklar, isteğe bağlı üstel
r8	Kayar noktalı sayı, büyüklük < = 1.7976931348623E+308, > = 2.225E-307 , `0..`9basamakları kullanılarak ifade edilir, isteğe bağlı işaret, isteğe bağlı kesirli basamaklar, isteğe bağlı üstel

Bir dizgi değeri boşluk içerebilir. Bir dizgi değerinde aşağıdaki kaçış dizilerini kullanmanız gerekir:

- & karakteri için`&`
- < karakteri için`<`

Aşağıdaki kaçış dizilerini kullanabilirsiniz, ancak bunlar gerekli değildir:

- > karakteri için`>`
- ' karakteri için`&apos;`
- " karakteri için`&quot;`

İlgili MQMD alanlarıyla birlikte JMS alanları ve özellikleri
Bu çizelgeler, JMS üstbilgi alanları, JMS özellikleri ve JMS sağlayıcıya özgü özellikler ile eşdeğeri MQMD alanlarını gösterir.

Çizelge 110 sayfa 784 , JMS üstbilgi alanlarını listeler ve Çizelge 111 sayfa 784 , doğrudan MQMD alanlarıyla eşlenen JMS özelliklerini listeler. Çizelge 112 sayfa 784 Sağlayıcıya özgü özellikleri ve eşlendikleri MQMD alanlarını listeler.

Çizelge 110. JMS üstbilgi alanları MQMD alanlarıyla eşleniyor

JMS üstbilgi alanı	**Java tipi**	**MQMD alanı**	**C tipi**
JMSDeliveryMode	int	Kalıcılık	MQLONG
JMSSüresi	uzun	Son kullanma tarihi	MQLONG
JMSönceliği	int	Öncelik	MQLONG
JMSMessageID	Dizgi	MsgID	MQBYTE24
JMSTimestamp	uzun	PutDate PutTime	MQCHAR8 MQCHAR8
JMSCorrelationID	Dizgi	CorrelId	MQBYTE24

Çizelge 111. MQMD alanlarıyla JMS özellikleri eşlemesi

JMS özelliği	**Java tipi**	**MQMD alanı**	**C tipi**
JMSXUserID	Dizgi	UserIdentifier	MQCHAR12
JMSXAppID	Dizgi	PutApplAdı	MQCHAR28
JMSXDeliveryCount	int	BackoutCount	MQLONG
JMSXGroupID	Dizgi	GroupId	MQBYTE24
JMSXGroupSeq	int	MsgSeqNumarası	MQLONG

Çizelge 112. JMS sağlayıcısına özgü özellikler-MQMD alanlarıyla eşleniyor

JMS sağlayıcısına özgü özellik	**Java tipi**	**MQMD alanı**	**C tipi**
JMS_IBM_Report_Exception	int	Rapor	MQLONG
JMS_IBM_Report_Son Kullanma	int	Rapor	MQLONG
JMS_IBM_Report_COA	int	Rapor	MQLONG
JMS_IBM_Report_COD	int	Rapor	MQLONG

Çizelge 112. JMS sağlayıcısına özgü özellikler-MQMD alanlarıyla eşleniyor (devamı var)

JMS sağlayıcısına özgü özellik	Java tipi	MQMD alanı	C tipi
JMS_IBM_Report_PAN	int	Rapor	MQLONG
JMS_IBM_Report_NAN	int	Rapor	MQLONG
JMS_IBM_Report_Pass_Msg_ID	int	Rapor	MQLONG
JMS_IBM_Report_Pass_Correl_ID	int	Rapor	MQLONG
JMS_IBM_Report_Disard_Msg	int	Rapor	MQLONG
JMS_IBM_MsgType	int	MsgType	MQLONG
JMS_IBM_Geri	int	Geribildirim	MQLONG
JMS_IBM_Biçimi	Dizgi	Biçim"1" sayfa 785	MQCHAR8
JMS_IBM_PutApplTipi	int	PutApplTipi	MQLONG
JMS_IBM_kodlama	int	Kodlama	MQLONG
JMS_IBM_Character_Set	Dizgi	CodedCharacterSetId"2" sayfa 785	MQLONG
JMS_IBM_PutDate	Dizgi	PutDate	MQCHAR8
JMS_IBM_PutTime	Dizgi	PutTime	MQCHAR8
JMS_IBM_Last_Msg_In_Group	boole	MsgFlags	MQLONG

Not:

1. JMS_IBM_Format, ileti gövdesinin biçimini temsil eder. Bu, uygulamanın JMS_IBM_Format özelliğini ayarlayarak (8 karakter sınırı olduğunu unutmayın) ya da JMS ileti tipine uygun ileti gövdesinin WebSphere MQ biçimine varsayılan olarak tanımlanabilirler. JMS_IBM_Format, yalnızca ileti RFH ya da RFH2 bölümleri içermiyorsa MQMD Biçimi alanıyla eşlenir. Tipik bir iletide, ileti gövdesinden hemen önce gelen RFH2 ' nin biçim alanıyla eşlenir.
2. JMS_IBM_Character_Set özellik değeri, sayısal CodedCharacterSetId değeri için Java karakter takımı eşdeğerini içeren bir Dize değeridir. MQMD alanı CodedCharacterSetId , JMS_IBM_Character_Set özelliği tarafından belirtilen Java karakter kümesi dizgisinin eşdeğerini içeren sayısal bir değerdir.

JMS alanlarının WebSphere MQ alanlarında eşlenmesi (giden iletiler)
Bu çizelgeler, JMS üstbilgisi ve özellik alanlarının, gönderme () ya da yayınlama () sırasında MQMD ve MQRFH2 alanlarına nasıl eşlendiğini gösterir.

Çizelge 113 sayfa 786 , JMS üstbilgi alanlarının gönderme () ya da yayınlama () sırasında MQMD/ RFH2 alanlarına nasıl eşlendiğini gösterir. Çizelge 114 sayfa 786 JMS özelliklerinin gönderme () ya da yayınlama () sırasında MQMD/RFH2 alanlarına nasıl eşlendiğini gösterir. Çizelge 115 sayfa 786 JMS sağlayıcısının özel özelliklerinin gönderme () ya da yayınlama () sırasında MQMD alanlarıyla nasıl eşlendiğini gösterir,

İleti nesnesine göre ayarlanmış olarak işaretlenmiş alanlar için, iletilen değer, gönderme () ya da yayınlama () işleminden hemen önce JMS iletisinde tutulan değerdir. JMS iletisinde değer, işlem tarafından değiştirilmeden bırakılır.

Gönderme Yöntemine Göre Ayarla işaretli alanlar için, gönderme () ya da yayınlama () gerçekleştirildiğinde (JMS iletisinde tutulan herhangi bir değer yoksayılır) bir değer atanır. JMS iletindeki değer, kullanılan değeri göstermek üzere güncelleştirilir.

Alma olarak işaretlenen alanlar iletilmez ve gönderilerek () ya da yayınla () ile değiştirilmeden bırakılır.

Çizelge 113. Giden ileti alanı eşlemesi

JMS üstbilgi alanı adı	İletim için kullanılan MQMD alanı	Üstbilgi	Ayarlayan
JMSHedef		MQRFH2	Yöntemi Gönder
JMSDeliveryMode	Kalıcılık	MQRFH2	Yöntemi Gönder
JMSSüresi	Son kullanma tarihi	MQRFH2	Yöntemi Gönder
JMSönceliği	Öncelik	MQRFH2	Yöntemi Gönder
JMSMessageID	MsgID		Yöntemi Gönder
JMSTimestamp	PutDate/PutTime		Yöntemi Gönder
JMSCorrelationID	CorrelId	MQRFH2	İleti Nesnesi
JMSReplyTo	ReplyToQ/ReplyToQMgr	MQRFH2	İleti Nesnesi
JMSType		MQRFH2	İleti Nesnesi
JMSRedird			Yalnızca alma

Not:

1. MQMD alanı CodedCharacterSetId , JMS_IBM_Character_Set özelliği tarafından belirtilen Java karakter kümesi dizgisinin eşdeğerini içeren sayısal bir değerdir.

Çizelge 114. Giden ileti JMS özelliği eşlemesi

JMS özellik adı	İletim için kullanılan MQMD alanı	Üstbilgi	Ayarlayan
JMSXUserID	UserIdentifier		Yöntemi Gönder
JMSXAppID	PutApplAdı		Yöntemi Gönder
JMSXDeliveryCount			Yalnızca alma
JMSXGroupID	GroupId	MQRFH2	İleti Nesnesi
JMSXGroupSeq	MsgSeqNumarası	MQRFH2	İleti Nesnesi

Çizelge 115. Giden ileti JMS sağlayıcısına özgü özellik eşlemesi

JMS sağlayıcısı-özel özellik adı	İletim için kullanılan MQMD alanı	Üstbilgi	Ayarlayan
JMS_IBM_Report_Exception	Rapor		İleti Nesnesi
JMS_IBM_Report_Son Kullanma	Rapor		İleti Nesnesi
JMS_IBM_Report_COA/COD	Rapor		İleti Nesnesi
JMS_IBM_Report_NAN/PAN	Rapor		İleti Nesnesi
JMS_IBM_Report_Pass_Msg_ID	Rapor		İleti Nesnesi
JMS_IBM_Report_Pass_Correl_ID	Rapor		İleti Nesnesi
JMS_IBM_Report_Disard_Msg	Rapor		İleti Nesnesi
JMS_IBM_MsgType	MsgType		İleti Nesnesi
JMS_IBM_Geri	Geribildirim		İleti Nesnesi

Çizelge 115. Giden ileti JMS sağlayıcısına özgü özellik eşlemesi (devamı var)			
JMS sağlayıcısı-özel özellik adı	**İletim için kullanılan MQMD alanı**	**Üstbilgi**	**Ayarlayan**
JMS_IBM_Biçimi	Biçim		İleti Nesnesi
JMS_IBM_PutApplTipi	PutApplTipi		Yöntemi Gönder
JMS_IBM_kodlama	Kodlama		İleti Nesnesi
JMS_IBM_Character_Set	CodedCharacterSetId		İleti Nesnesi
JMS_IBM_PutDate	PutDate		Yöntemi Gönder
JMS_IBM_PutTime	PutTime		Yöntemi Gönder
JMS_IBM_Last_Msg_In_Group	MsgFlags		İleti Nesnesi

Gönderme () ya da yayınlama () sırasında JMS üstbilgisi alanları eşleniyor
Bu notlar, gönderme () ya da yayınlama () sırasında JMS alanlarının eşlemeleriyle ilgilidir.

JMSDestination- MQRFH2
Bu, hedef nesnenin belirgin özelliklerini diziselleştiren bir dizgi olarak saklanır; böylece, alan bir JMS, eşdeğer bir hedef nesneyi yeniden oluşturabilirler. MQRFH2 alanı URI olarak kodlanır (URI gösterimine ilişkin ayrıntılar için “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844 ' a bakın).

JMSReplyTo - MQMD.ReplyToQ, ReplyToQMgr, MQRFH2
Kuyruk adı MQMD.ReplyToQ alanı ve kuyruk yöneticisi adı ReplyToQMgr alanlarına kopyalanır. Hedef uzantı bilgileri (hedef nesnede saklanan diğer yararlı ayrıntılar), MQRFH2 alanına kopyalanır. MQRFH2 alanı URI olarak kodlanır (URI gösterimine ilişkin ayrıntılar için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844).

JMSDeliveryMode - MQMD.Persistence
The JMSDeliveryMode value is set by the send() or publish() Method or MessageProducer, unless the Destination Object overrides it. JMSDeliveryMode değeri, MQMD.Persistence alanı aşağıdaki gibi olur:

- PERSISTENT JMS değeri MQPER_PERSISTENT ile eşdeğerdir
- JMS değeri NON_PERSISTENT, MQPER_NOT_PERSISTENT ile eşdeğerdir

MQQueue persistence özelliği WMQConstants.WMQ_PER_QDEFolarak ayarlanmadıysa, teslim kipi değeri de MQRFH2içinde kodlanır.

JMSExpiration ile MQMD.Expiry, MQRFH2
JMSExpiration, süre bitimi (geçerli zamanın ve etkin zamanın toplamını) saklar, ancak MQMD, zaman zaman yaşamasını sağlar. Ayrıca, JMSExpiration milisaniyedir, ancak MQMD.Expiry saniyenin onda biri cinsinden ifade edilir.

- Gönderme () yöntemi, yaşamak için sınırsız bir zaman belirtiyorsa, MQMD.Expiry MQE_UNSıNSıSıA olarak ayarlanır ve MQRFH2' da JMSExpiration kodlanmaz.
- If the send() method sets a time to live that is less than 214748364.7 seconds (about 7 years), the time to live is stored in MQMD.Expiry, and the expiration time (in milliseconds), is encoded as an i8 value in the MQRFH2.
- Gönderme () yöntemi bir saati 214748364.7 saniyeden daha uzun bir süre ayarlarsa, MQMD.Expiry MQEI_UNESSıNA olarak ayarlıdır. Milisaniye cinsinden gerçek süre bitimi, MQRFH2içinde bir i8 değeri olarak kodlanır.

JMSPriority- MQMD.Priority
JMSPriority değerini (0-9), MQMD öncelik değerine (0-9) doğrudan eşleyin. JMSPriority varsayılan olmayan bir değere ayarlandıysa, öncelik düzeyi MQRFH2içinde de kodlanır.

MQMD.MessageID' danJMSMessageID
JMS ' den gönderilen tüm iletiler, WebSphere MQtarafından atanan benzersiz ileti tanıtıcılarına sahiptir. Atanan değer, MQMD.MessageId alanı, MQPUT çağrısından sonra ve JMSMessageID alanında uygulamaya geri geçirilir. WebSphere MQ messageId , 24 baytlık ikili bir değerdir, ancak

JMSMessageID bir dizgidir. JMSMessageID , ikili messageId değerinden 48 onaltılı karakterden oluşan bir sıraya dönüştürüldü; önek olarak karakter tanıtıcısı:. JMS, ileti tanıtıcıları üretmesini geçersiz kılmak için ayarlanabilen bir ipucu sağlar. Bu ipucu yok sayılır ve tüm durumlarda benzersiz bir tanıtıcı atanır. Bir gönderinin () üzerine yazılmadan önce JMSMessageId alanına ayarlanan herhangi bir değer üzerine yazılır.

Gerekiyorsa, MQMD.MessageID, bunu, "JMS uygulaması için bir WebSphere MQ sınıflarından ileti tanımlayıcısının okunması ve yazılması" sayfa 859içinde açıklanan WebSphere MQ JMS uzantılarından biriyle yapabilirsiniz.

JMSTimestamp- MQRFH2
Bir gönderme sırasında, JMSTimestamp alanı JVM ' nin saatine göre ayarlanır. Bu değer MQRFH2' ye ayarlanır. Bir gönderinin () üzerine yazılmadan önce JMSTimestamp alanına ayarlanan herhangi bir değerin üzerine yazılır. Ayrıca, JMS_IBM_PutDate ve JMS_IBM_PutTime özelliklerine de bakın.

JMSType- MQRFH2
Bu dizgi, MQRFH2 mcd.Type alanına ayarlanır. URI biçiminde olması durumunda, mcd.Set ve mcd.Fmt alanlarını da etkileyebilir. Ayrıca bkz. " WebSphere Event Broker ya da WebSphere Message Broker aracısıyla gerçek zamanlı bir bağlantı kullanılması" sayfa 886.

JMSCorrelationID - MQMD.CorrelId, MQRFH2
JMSCorrelationID aşağıdakilerden birini tutabilir:

Sağlayıcıya özgü ileti tanıtıcısı
This is a message identifier from a message previously sent or received, and so should be a string of 48 lowercase hexadecimal digits that are prefixed with *Tanıtıcı:* The prefix is removed, the remaining characters are converted into binary, and then they are set into the MQMD.CorrelId field. No CorrelId value is encoded in the MQRFH2.

Sağlayıcı-yerel bayt [] değer
The value is copied into the MQMD.CorrelId field - padded with nulls, or truncated to 24 bytes if necessary. No CorrelId value is encoded in the MQRFH2.

Uygulamaya özgü bir dizgi
Değer MQRFH2' ye kopyalanır. Dizginin ilk 24 baytı UTF8 biçiminde, MQMD.CorrelID.

JMS özellik alanlarının eşlenmesi
Bu notlar, WebSphere MQ iletilerindeki JMS özelliği alanlarının eşlenmesine başvurur.

JMSXUserID from MQMD UserIdentifier
JMSXUserID , gönderme çağrısından geri dönüş olarak ayarlandı.

JMSXAppID from MQMD PutApplName
JSMXAppID , gönderme çağrısından geri dönüş olarak ayarlandı.

JMSXGroupID - MQRFH2 (noktadan noktaya iletişim)
Noktadan noktaya iletişim iletileri için, JMSXGroupID , MQMD GroupID alanına kopyalanır. JMSXGroupID önek tanıtıcısı ile başlıyorsa, ikili olarak dönüştürülür. Tersi durumda, bu dizgi bir UTF8 dizgisi olarak kodlanır. Gerekiyorsa, değer doldurulur ya da kesilir ve 24 byte uzunluğunda bir değer kesilir. MQMF_MSG_IN_GROUP işareti ayarlıdır.

JMSXGroupID - MQRFH2 (yayınlama/abone olma)
Yayınlama/abone olma iletileri için, JMSXGroupID bir dizgi olarak MQRFH2 içine kopyalanır.

JMSXGroupSeq MQMD MsgSeqSayı (noktadan noktaya iletişim)
Noktadan noktaya iletişim iletileri için, JMSXGroupSeq , MQMD MsgSeqSayı alanına kopyalanır. MQMF_MSG_IN_GROUP işareti ayarlıdır.

JMSXGroupSeq MQMD MsgSeqNumarası (yayınlama/abone olma)
Yayınlama/abone olma iletileri için, JMSXGroupSeq , MQRFH2 içine i4olarak kopyalanır.

JMS sağlayıcısına özgü alanlar eşleniyor
Aşağıdaki notlarda, JMS sağlayıcısına özgü alanların IBM WebSphere MQ iletilerine eşlenmesine ilişkin bilgiler yer alir.

JMS_IBM_Report_ < ad > MQMD Raporuna
Bir JMS uygulaması, aşağıdaki JMS_IBM_Report_XXX özelliklerini kullanarak MQMD Rapor seçeneklerini ayarlayabilir. Tek MQMD, birkaç JMS_IBM_Report_XXX özellikleriyle eşlenir. Uygulama, bu özelliklerin değerini standart IBM WebSphere MQ MQRO_ sabitlerine (com.ibm.mq.MQCiçinde bulunur) ayarlamalıdır. Örneğin, COD 'yi tam verilerle istemek için uygulama JMS_IBM_Report_COD' yi CMQC.MQRO_COD_WITH_FULL_DATA.

JMS_IBM_Report_Exception

MQRO_EXCEPTION ya da
MQRO_EXCEPTION_WITH_DATA ya da
MQRO_EXCEPTION_WITH_FULL_DATA

JMS_IBM_Report_Son Kullanma

MQRO_EXPIRATION ya da
MQRO_EXPIRATION_WITH_DATA ya da
MQRO_EXPIRATION_WITH_FULL_DATA

JMS_IBM_Report_COA

MQRO_COA ya da
MQRO_COA_WITH_DATA ya da
MQRO_COA_WITHL_FULL_DATA

JMS_IBM_Report_COD

MQRO_COD ya da
MQRO_COD_WITH_DATA ya da
MQRO_COD_WITH_FULL_DATA

JMS_IBM_Report_PAN
MQRO_PAN

JMS_IBM_Report_NAN
MQRO_NAN

JMS_IBM_Report_Pass_Msg_ID
MQRO_PASS_MSG_ID

JMS_IBM_Report_Pass_Correl_ID
MQRO_PASS_COREL_ID

JMS_IBM_Report_Disard_Msg
MQRO_DISCARD_MSG

JMS_IBM_MsgType ile MQMD MsgTypeiçin
Değer, doğrudan MQMD MsgType' ta eşlenir. Uygulama, JMS_IBM_MsgTypeiçin belirtilmiş bir değer ayarlamadıysa, varsayılan değer kullanılır. Bu varsayılan değer aşağıdaki gibi belirlenir:

- JMSReplyTo bir IBM WebSphere MQ kuyruk hedefi olarak ayarlandıysa, MSGType MQMT_REQUEST değerine ayarlanır.
- JMSReplyTo belirlenmezse ya da IBM WebSphere MQ kuyruk hedefi dışında bir değere ayarlandıysa, MsgType MQMT_DATAGRAM değerine ayarlanır.

JMS_IBM_FeedFeedback-MQMD
Değer, doğrudan MQMD geribildirimiyle eşlenir

JMS_IBM_Biçimi-MQMD Biçiminde
Değer, doğrudan MQMD biçimiyle eşlenir.

JMS_IBM_kodlamasını MQMD Kodlamasına
Ayarlanmışsa, bu özellik, Hedef Kuyruk ya da Konu 'nın sayısal kodlamasını geçersiz kılar.

JMS_IBM_Character_Set to MQMD CodedCharacterSetId
Ayarlanırsa, bu özellik, Hedef Kuyruk ya da Konu 'nın kodlanmış karakter kümesi özelliğini geçersiz kılar.

JMS_IBM_PutDate , MQMD PutDate' den
Bu özelliğin değeri, gönderme sırasında, doğrudan MQMD ' deki PutDate alanından ayarlanır. Bir gönderinin üzerine yazılmadan önce, JMS_IBM_PutDate özelliğinde ayarlanan herhangi bir değer. Bu alan, YYYYAAGG ' nin IBM WebSphere MQ Tarih biçiminde, sekiz karakterlik bir dizilimdir. Bu özellik, iletinin kuyruk yöneticisine göre ne zaman konabileceğini belirlemek için JMS_IBM_PutTime özelliğiyle kullanılabilir.

JMS_IBM_PutTime , MQMD PutTime' den
Bu özelliğin değeri, gönderme sırasında, doğrudan MQMD ' deki PutTime alanından ayarlanır. Bir gönderinin üzerine yazılmadan önce, JMS_IBM_PutTime özelliğinde ayarlanan herhangi bir değer. This field is a String of eight characters, in the IBM WebSphere MQ Time format of HHMMSSTH. Bu özellik, iletinin kuyruk yöneticisine göre nereye konacağı zamanı belirlemek için JMS_IBM_PutDate özelliğiyle birlikte kullanılabilir.

JMS_IBM_Last_Msg_In_Group-MQMD MsgFlags
Noktadan noktaya ileti sistemi için bu Boole değeri, MQMD MsgFlags alanındaki MQMF_LAST_MSG_IN_GROUP işaretiyle eşlenir. Olağan bir şekilde, bu iletinin bir grupta son olduğunu gösteren eski bir IBM WebSphere MQ uygulamasına göstermek için JMSXGroupID ve JMSXGroupSeq özellikleriyle kullanılır. Yayınlama/abone olma ileti alışverişi için bu özellik yok sayılır.

WebSphere MQ alanlarının JMS alanlarıyla eşlenmesi (gelen iletiler)
Bu çizelgeler, JMS üstbilgisi ve özellik alanlarının, get () ya da alma () sırasında MQMD ve MQRFH2 alanlarına nasıl eşlendiğini gösterir.

Çizelge 116 sayfa 790 , JMS üstbilgi alanlarının get () ya da alma () zamanındaki MQMD/MQRFH2 alanlarına nasıl eşlendiğini gösterir. Çizelge 117 sayfa 791 JMS özellik alanlarının get () ya da alma () zamanındaki MQMD/MQRFH2 alanlarına nasıl eşlendiğini gösterir. Çizelge 118 sayfa 791 JMS sağlayıcısına özgü özelliklerin nasıl eşlendiğini gösterir.

Çizelge 116. Gelen ileti JMS üstbilgi alanı eşlemesi

JMS üstbilgi alanı adı	**MQMD alanı alındı**	**MQRFH2 alanı şu alandan alındı:**
JMSHedef		jms.Dst ya da mqps.Top"1" sayfa 791
JMSDeliveryMode	Kalıcılık"2" sayfa 791	jms.Dlv"2" sayfa 791
JMSSüresi		jms.Exp
JMSönceliği	Öncelik	
JMSMessageID	MsgID	
JMSTimestamp	PutDate"2" sayfa 791 PutTime"2" sayfa 791	jms.Tms"2" sayfa 791
JMSCorrelationID	CorrelId"2" sayfa 791	jms.Cid"2" sayfa 791
JMSReplyTo	ReplyToQ"2" sayfa 791 ReplyToQMgr"2" sayfa 791	jms.Rto"2" sayfa 791
JMSType		mcd.Type, mcd.Set, mcd.Fmt
JMSRedird	BackoutCount	

Çizelge 116. Gelen ileti JMS üstbilgi alanı eşlemesi (devamı var)

JMS üstbilgi alanı adı	MQMD alanı alındı	MQRFH2 alanı şu alandan alındı:

Not:

1. Hem jms.Dst hem de mqps.Top ayarlandıysa, jms.Dst içindeki değer kullanılır.
2. MQRFH2 'den ya da MQMD' den alınan değerlere sahip olan özellikler için, her ikisi de kullanılabilir durumda ise, MQRFH2 ' deki ayar kullanılır.
3. JMS_IBM_Character_Set özellik değeri, sayısal CodedCharacterSetId değeri için Java karakter takımı eşdeğerini içeren bir Dize değeridir.

Çizelge 117. Gelen ileti özelliği eşlemesi

JMS özellik adı	MQMD alanı alındı	MQRFH2 alanı şu alandan alındı:
JMSXUserID	UserIdentifier	
JMSXAppID	PutApplAdı	
JMSXDeliveryCount	BackoutCount	
JMSXGroupID	GroupId“1” sayfa 791	jms.Gid“1” sayfa 791
JMSXGroupSeq	MsgSeqNumara“1” sayfa 791	jms.Seq“1” sayfa 791

Not:

1. MQRFH2 'den ya da MQMD' den alınan değerlere sahip olan özellikler için, her ikisi de kullanılabilir durumda ise, MQRFH2 ' deki ayar kullanılır. Bu özellikler yalnızca MQMF_MSG_IN_GROUP ya da MQMF_LAST_MSG_IN_GROUP ileti işaretleri ayarlandıysa, MQMD değerlerinden ayarlanır.

Çizelge 118. Gelen ileti sağlayıcıya özgü JMS özelliği eşlemesi

JMS özellik adı	MQMD alanı alındı	MQRFH2 alanı şu alandan alındı:
JMS_IBM_Report_Exception	Rapor	
JMS_IBM_Report_Son Kullanma	Rapor	
JMS_IBM_Report_COA	Rapor	
JMS_IBM_Report_COD	Rapor	
JMS_IBM_Report_PAN	Rapor	
JMS_IBM_Report_NAN	Rapor	
JMS_IBM_Report_ Pass_MsgID	Rapor	
JMS_IBM_Report_Pass_Correl_ID	Rapor	
JMS_IBM_Report_Disard_Msg	Rapor	
JMS_IBM_MsgType	MsgType	
JMS_IBM_Geri	Geribildirim	
JMS_IBM_Biçimi	Biçim	
JMS_IBM_PutApplTipi	PutApplTipi	

Çizelge 118. Gelen ileti sağlayıcıya özgü JMS özelliği eşlemesi (devamı var)		
JMS özellik adı	**MQMD alanı alındı**	**MQRFH2 alanı şu alandan alındı:**
JMS_IBM_Encoding [“1” sayfa 792]	Kodlama	
JMS_IBM_Character_Set [“1” sayfa 792]	CodedCharacterSetId	
JMS_IBM_PutDate	PutDate	
JMS_IBM_PutTime	PutTime	
JMS_IBM_Last_Msg_In_Group	MsgFlags	
1. Yalnızca gelen ileti bir Bytes İletisi ise ayarlanır.		

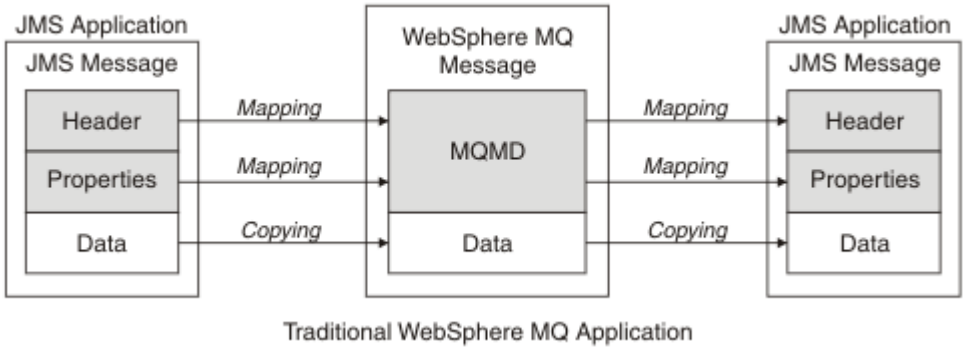
JMS uygulaması ile geleneksel bir WebSphere MQ uygulaması arasında ileti alışverişi yapma
This topic describes what happens when a JMS application exchanges messages with a traditional WebSphere MQ application that cannot process the MQRFH2 header.

. Şekil 129 sayfa 792 , eşlemeyi gösterir.

Yönetici, JMS uygulamasının hedef *MQ*değerine ayarlayarak geleneksel bir WebSphere MQ uygulaması ile iletişim kurduğunu gösterir. Bu, MQRFH2 üstbilgisinin üretilmemesine neden olmadığını gösterir. Bu işlem yapılmazsa, alma uygulaması MQRFH2 üstbilgisini işleyebilmelidir.

Geleneksel bir WebSphere MQ uygulamasında hedeflenen JMS 'den MQMD' ye eşleme, JMS 'den hedeflenen MQMD' ye eşleme ile aynı. JMS uygulamasında hedeflenen eşleme. JMS için WebSphere MQ sınıfları, MQMD `Format` alanı MQFMT_RFH2dışında bir WebSphere MQ iletisi alırsa, JMS olmayan bir uygulamadan veri alınıyor. Biçim MQFMT_STRING ise, ileti bir JMS metin iletisi olarak alınır. Tersi durumda, JMS byte iletisi olarak alınır. MQRFH2olmadığı için, yalnızca MQMD ' de iletilen JMS özellikleri geri yüklenebilir.

JMS için WebSphere MQ sınıfları MQRFH2 üstbilgisi olmayan bir ileti alırsa, iletinin JMSReplyTo üstbilgi alanından türetilmiş olan Kuyruk ya da Konu nesnesinin TARGCLIENT özelliği varsayılan olarak MQ olarak ayarlanır. Bu, kuyruğa ya da konuya gönderilen bir yanıt iletisinin MQRFH2 üstbilgisine de sahip olmadığı anlamına gelir. Bir yanıt iletisinde MQRFH2 üstbilgisi dahil olmak üzere bu davranışı, özgün iletinin bir MQRFH2 üstbilgisi varsa, bağlantı üreticisinin TARGCLIENTMATCHESLE özelliğini NO olarak ayarlayarak değiştirebilirsiniz.



Şekil 129. JMS iletilerinin MQRFH2 üstbilgisi içermeyen WebSphere MQ iletilerine nasıl dönüştürülebiliyor

JMS ileti gövdesi
Bu konu, ileti gövdesinin kodlamasıyla ilgili bilgileri içerir. Kodlama, JMS iletisinin tipine bağlıdır.

ObjectMessage
ObjectMessage , Java Runtime tarafından olağan biçimde diziselleştirilmiş bir nesnedir.

TextMessage

TextMessage , kodlanmış bir dizgidir. Giden bir ileti için, dizgi hedef nesne tarafından verilen karakter kümesiyle kodlanır. Bu, varsayılan değer olarak UTF8 kodlamasına ayarlanır (UTF8 kodlaması, iletinin ilk karakteriyle başlar; başlangıçta uzunluk alanı yoktur). Ancak, JMS için WebSphere MQ sınıfları tarafından desteklenen başka herhangi bir karakter kümesini belirtmek mümkün olabilir. Bu tür karakter kümeleri daha çok JMS dışı bir uygulamaya ileti gönderdiğinizde kullanılır.

Karakter kümesi çift baytlık bir kümese (UTF16da içinde olmak üzere), hedef nesnenin tamsayı kodlama belirtimi byte 'ın sırasını belirler.

Gelen ileti, iletide belirtilen karakter kümesi ve kodlamayla yorumlanır. Bu belirtimler son WebSphere MQ üstbilgisinde (ya da üstbilgi yoksa MQMD) bulunur. JMS iletileri için, son üstbilgi genellikle MQRFH2' dir.

BytesMessage

BytesMessage varsayılan olarak, JMS 1.0.2 belirtimi ve ilişkili Java belgeleri tarafından tanımlanan bir bayt dizisi demektir.

Uygulamanın kendisi tarafından birleştirilen bir giden ileti için, hedef nesnenin kodlama özelliği, iletide yer alan tamsayı ve kayar noktalı alanların kodlamaları geçersiz kılmak için kullanılabilir. Örneğin, kayan noktalı değerlerin IEEE biçimi yerine S/390 içinde saklanabileceğini isteyebilirsiniz.)

Gelen bir ileti, iletinin kendisinde belirtilen sayısal kodlama kullanılarak yorumlanır. Bu belirtim son WebSphere MQ üstbilgisinde (ya da üstbilgi yoksa MQMD) bulunur. JMS iletileri için, son üstbilgi genellikle MQRFH2' dir.

Bir BytesMessage (BytesMessage) alınırsa ve değiştirilmeden yeniden gönderilirse, gövdesi, alındığı için bayt for byte 'ı iletir. Hedef nesnenin kodlama özelliğinin gövde üzerinde etkisi yok. BytesMessage içinde açık olarak gönderilebilen tek dize benzeri varlık, bir UTF8 dizesidir. Bu, Java UTF8 biçiminde kodlanır ve 2 baytlık bir alanla başlar. Hedef nesnenin karakter kümesi özelliğinin giden bir BytesMessageöğesinin kodlaması üzerinde hiçbir etkisi yoktur. Gelen bir WebSphere MQ iletisinde karakter kümesi değeri, bu iletinin yorumunda bir JMS BytesMessageolarak bir etkisi yoktur.

Java dışı uygulamaların Java UTF8 kodlamasını tanıması olası değildir. Bu nedenle, bir JMS uygulamasının metin verisi içeren bir BytesMessage göndermesini sağlamak için uygulamanın kendi dizelerini bayt dizilerine dönüştürmesi ve bu bayt dizilerini BytesMessage' a yazması gerekir.

MapMessage

MapMessage , şu şekilde kodlanan XML ad/tip/değer üçüzleri içeren bir dizgidir:

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

Burada `datatype` , Çizelge 109 sayfa 783içinde listelenen veri tiplerinden biridir. Varsayılan veri türü `string`' dir ve bu nedenle dizgi öğeleri için `dt="string"` özniteliği atlanır.

Bir eşlem iletisinin gövesini oluşturan XML dizesini kodlamak ya da yorumlamak için kullanılan karakter kümesi, bir metin iletisi için geçerli olan kurallara göre belirlenir.

Versions of WebSphere MQ classes for JMS earlier than Version 5.3 encoded the body of a map message in the following format:

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

Version 5.3 and later versions of WebSphere MQ classes for JMS can interpret either format, but versions of WebSphere MQ classes for JMS earlier than Version 5.3 cannot interpret the current format.

If an application needs to send map messages to another application that is using a version of WebSphere MQ classes for JMS earlier than Version 5.3, the sending application must call the connection factory method `setMapNameStyle(WMQConstants.WMQ_MAP_NAME_STYLE_COMPATIBLE)` to specify that the map messages are sent in the previous format. Varsayılan olarak, tüm eşlem iletileri yürürlükteki biçimde gönderilir.

StreamMessage
Bir StreamMessage , bir eşleme iletisi gibidir, ancak öğe adları olmadan:

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

Burada `datatype` , Çizelge 109 sayfa 783içinde listelenen veri tiplerinden biridir. Varsayılan veri türü `string`' dir ve bu nedenle dizgi öğeleri için `dt="string"` özniteliği atlanır.

StreamMessage gövdeyi oluşturan XML dizesini kodlamak ya da yorumlamak için kullanılan karakter kümesi, TextMessageiçin geçerli olan kurallardan sonra belirlenir.

MQRFH2.format alanı aşağıdaki gibi ayarlanır:

MQFMT_NONE
ObjectMessage, BytesMessageiçin ya da gövdesi olmayan iletiler için.

MQFMT_STRING
TextMessage, StreamMessageya da MapMessageiçin.

JMS ileti dönüştürme

İletileri gönderirken ve alırken JMS ' de ileti verisi dönüştürme işlemi gerçekleştirilir. WebSphere MQ , çoğu veri dönüştürme işlemini otomatik olarak gerçekleştirir. JMS uygulamaları arasında bir ileti aktarırken metin ve sayısal verileri dönüştürür. Bir JMS uygulaması ile bir WebSphere MQ uygulaması arasında `JMSTextMessage` değiş tokacı değiş tokuş edildiğinde metin dönüştürüldü.

Daha karmaşık ileti alışverişleri yapmayı planlıyorsanız, aşağıdaki konular ilginizi çekmektedir. Karmaşık ileti alışverişi şunları içerir:

- Bir WebSphere MQ uygulaması ile JMS uygulaması arasında metin olmayan iletilerin aktarılması.
- Metin verilerinin bayt biçiminde değiş tokuş edilmesine neden olur.
- Uygulamandaki metnin dönüştürülmesi.

JMS ileti verileri

İki JMS uygulaması arasında bile metin ve sayısal verileri uygulamalar arasında değiş tokuş etmek için veri dönüştürme gereklidir. Metin ve sayıların iç gösterimi kodlanmalıdır, bu nedenle bir iletide aktarılabilirler. Kodlama, sayıların ve metnin nasıl temsil edildiği hakkında bir karar sağlar. WebSphere MQ JMS iletilerindeki metin ve sayıların kodlamasını yönetir (`JMSObjectMessage`edışında, bkz. “JMSObjectMessage” sayfa 801). Üç ileti özniteliğini kullanır. Üç öznitelik şunlardır: `CodedCharacterSetId`, `Encoding`, ve `Format`.

Bu üç ileti özniteliği genellikle JMS üstbilgisinde, `MQRFH2`' da bir JMS iletisinin alanlarında saklanır. İleti tipi, `JMS` ileti tipi yerine bir `MQ`ise, öznitelikler ileti tanımlayıcısında (`MQMD`) saklanır. Öznitelikler JMS ileti verilerini dönüştürmek için kullanılır. JMS ileti verileri, WebSphere MQ iletisinin ileti verileri kısmında aktarılır.

JMS iletisi özellikleri

İleti bir `MQRFH2`olmadan gönderilmediyse, `JMS_IBM_CHARACTER_SET`gibi JMS ileti özellikleri, JMS iletisinin `MQRFH2` üstbilgi bölümünde değiştirilir. Yalnızca `JMSTextMessage` ve `JMSBytesMessage` , `MQRFH2`olmadan gönderilebilir. Bir JMS özelliği, ileti tanımlayıcısında WebSphere MQ ileti özelliği olarak saklanırsa, `MQMD`, `MQMD` dönüştürmesinin bir parçası olarak dönüştürülür. Bir JMS özelliği `MQRFH2`'

de saklanırsa, MQRFH2.NameValueCCSIDtarafından belirtilen karakter kümesinde saklanır. Bir ileti gönderildiğinde ya da alındığında, ileti özellikleri JVM ' deki iç gösterimlerine ve bunların iç gösterimlerine dönüştürülmektedir. Dönüştürme, ileti tanımlayıcısı ya da MQRFH2.NameValueCCSIDkarakter takımından ve karakter takımından olmalıdır. Sayısal veriler metne dönüştürülür.

JMS ileti dönüştürme

Aşağıdaki konularda, dönüştürme gerektiren daha karmaşık iletiler değiş tokuş etmeyi planlıyorsanız, yararlı olan örnekler ve görevler yer alır.

JMS ileti dönüştürme yaklaşımları
Bir dizi veri dönüştürme yaklaşımları JMS uygulama tasarımcılarına açıktır. Bu yaklaşımlar münhasır değildir; bazı uygulamalar bu yaklaşımların bir birleşimini kullanma olasılığının yüksek olduğu bir tür. Uygulamanız yalnızca metin alışverişi yapmışsa ya da yalnızca diğer JMS uygulamalarıyla ileti alışverişi yapmışsa, normalde veri dönüştürmeyi dikkate almayın. Veri dönüştürme işlemi sizin için otomatik olarak WebSphere MQtarafından gerçekleştirilir.

İleti dönüştürmenin nasıl yaklaşılacağı hakkında bir dizi soru sorabilirsiniz:

Mesaj dönüşümünü düşünmek için gerekli mi?
In some cases, such as JMS to JMS message transfers, and exchanging text messages with IBM WebSphere MQ programs, IBM WebSphere MQ performs the necessary conversions for you, automatically. Performans nedenlerine ilişkin veri dönüştürmeyi denetlemek isteyebilirsiniz ya da önceden tanımlanmış bir biçime sahip karmaşık iletiler değiş tokacı olabilir. Bu gibi durumlarda, ileti dönüşümünü anlamalı ve aşağıdaki konuları okumanız gerekir.

Ne tür bir dönüşüm var?
Aşağıdaki bölümlerde açıklanan dört ana dönüştürme türü vardır:

1. “JMS istemcisi veri dönüştürme” sayfa 795
2. “Uygulama verileri dönüştürme” sayfa 796
3. “Kuyruk yöneticisi verileri dönüştürme” sayfa 796
4. “İleti kanalı veri dönüştürme” sayfa 797

Dönüştürme gerçekleştirilmeli mi?
The section, “İleti dönüştürmesi için bir yaklaşım seçilmesi: alıcı iyi olur” sayfa 797, describes the usual approach of "alıcı iyi olur". "Alıcı iyi olur" , JMS veri dönüştürmesi için de geçerlidir.

JMS istemcisi veri dönüştürme

JMS istemcisi[4]Veri dönüştürme işlemi, Java primitives ve nesnelerinin bir JMS iletisinde gönderildiği bir JMS iletisinde bayt olarak dönüştürülmesini ve alındığında yeniden dönüştürme işlemi olduğunu yeniden sağlar. JMS istemcisi veri dönüştürmesi, JMSMessage sınıflarının yöntemlerini kullanır. Yöntemler, Çizelge 119 sayfa 799içinde JMSMessage sınıf tipine göre listelenir.

Okuma, alma, ayarlama ve yazma yöntemleri için, sayıların ve metinlerin iç JVM temsiline dönüştürme işlemi gerçekleştirilir. Dönüştürme işlemi, ileti gönderildiğinde ve alınan bir iletiyle ilgili okuma ya da alma yöntemlerinden herhangi biri çağrıldığında gerçekleştirilir.

Bir iletinin içeriğini yazmak ya da belirlemek için kullanılan kod sayfası ve sayısal kodlama, hedefin öznitelikleri olarak tanımlanır. Hedef kod sayfası ve sayısal kodlama yönetimsel olarak değiştirilebilir. Bir uygulama ayrıca, ileti içeriğini denetleyen ya da ileti içeriğini denetleyen ileti özelliklerini ayarlayarak, hedef kod sayfasını ve kodlamasını geçersiz kılabilir.

If you want to convert number encoding when a JMSBytesMessage message is sent to a destination that is not defined as Native encoding, you must set the message property JMS_IBM_ENCODING before sending the message. "Günlük nesnesini iyi kılar" örüntlerini izliyorsanız ya da JMS uygulamaları

[4] "JMS İstemcisi" , istemci ya da bağ tanımları kipinde çalışan JMS arabirimini gerçekleştiren JMS için WebSphere MQ sınıflarına gönderme yapar.

arasında ileti alışverişi yapıyorsanız, uygulamanın JMS_IBM_ENCODINGöğesini ayarlamaya gerek yoktur. Çoğu durumda Encoding özelliğini Nativeolarak bırakabilirsiniz.

JMSStreamMessage, JMSMapMessageve JMSTextMessage iletileri için, hedefin karakter kümesi tanıtıcı özellikleri kullanılır. Kod, metin biçiminde yazıldığı için kodlamada yoksayılır. JMS istemcisi uygulama programının, hedef karakter kümesi özelliğinin geçerli olması durumunda iletiyi göndermeden önce JMS_IBM_CHARACTER_SET ayarına sahip olması gerekmez.

Bir iletide verileri almak için, JMS iletisi okuma ya da alma yöntemlerini çağıran bir uygulamadır. Yöntemler, Java temel öğelerini ve nesnelerini doğru olarak yaratmak için önceki ileti üstbilgisinde tanımlanan kod sayfası ve kodlamaya gönderme yapıyor.

JMS istemcisi veri dönüştürmesi, bir JMS istemcisi ile başka bir JMS istemcisi arasında ileti alışverişi yapan çoğu JMS uygulamasının gereksinimlerini karşılar. Herhangi bir belirtik veri dönüştürmesi kodlamazsiniz. Genellikle, bir dosyaya metin yazılırken kullanılan java.nio.charset.Charset sınıfını kullanmaz. writeString ve setString yöntemleri, sizin için dönüştürmeyi yapar.

JMS istemcisi veri dönüştürmeye ilişkin ayrıntılar için bkz. “JMS istemcisi ileti dönüştürme ve kodlama” sayfa 807.

Uygulama verileri dönüştürme

A JMS client application can perform explicit character data conversion by using the java.nio.charset.Charset class; see the examples in Şekil 132 sayfa 800 and Şekil 133 sayfa 800 . Dizgi verileri byte olarak dönüştürülür (getBytes yöntemi kullanılarak) ve byte olarak gönderilir. Bayt dizisi ve Charsetkullanan bir String oluşturucusu kullanılarak, byte 'lar metne geri dönüştürülür. Karakter verileri, encode ve decode Charset yöntemleri kullanılarak dönüştürülür. Typically the message is sent or received as JMSBytesMessage, because the message part of a JMSBytesMessage does not contain anything other than the data written by the application[5]. You can also send and receive bytes using JMSStreamMessage, JMSMapMessage, or JMSObjectMessage.

Farklı kodlama biçimlerinde gösterilen sayısal veriler içeren byte kodlarının kodlanması ve kodunu çözmek için Java yöntemi yoktur. Sayısal veriler, sayısal JMSMessage okuma ve yazma yöntemleri kullanılarak otomatik olarak kodlanır ve kod çözücüdür. Okuma ve yazma yöntemleri, ileti verilerinin JMS_IBM_ENCODING özniteliğin değerini kullanır.

Uygulama verilerinin dönüştürülmesi için tipik bir kullanım, JMS istemcisinin JMS dışı bir uygulamadan bir biçimlendirilmiş ileti göndermesi ya da göndermesi durumunda olur. Biçimlendirilmiş ileti, veri alanlarının uzunluğuna göre düzenlenmiş metin, sayısal ve byte verilerini içerir. JMS dışı uygulama ileti biçimini "MQSTR "olarak belirtmediyse, ileti bir JMSBytesMessageolarak oluşturulur. JMSBytesMessage ' ta biçimlendirilmiş ileti verilerini almak için bir yöntem dizisi çağırmalısınız. Yöntemlerin, aynı sırayla çağrılması gerekir; bu yöntemler, alanlara veri yazıldığı sırayla yazılır. Alanlar sayısalsa, sayısal verilerin kodlamasını ve uzunluğunu bilmeniz gerekir. Alanlardan herhangi biri byte ya da metin verisi içeriyorsa, iletteki herhangi bir bayt veri uzunluğunu bilmeniz gerekir. Biçimlendirilmiş bir iletiyi, kullanımı kolay bir Java nesnesine dönüştürmenin iki yolu vardır.

1. Okuma yazma ve iletiyi yazma işlemi için, kayda karşılık gelen bir Java sınıfı oluşturun. Kayıttaki verilere erişim, sınıfın alma ve ayarlama yöntemleriyle birlikte olur.
2. com.ibm.mq.headers sınıfını genişleterek, kayıtla ilgili bir Java sınıfı oluşturun. Sınıftaki verilere erişim, formun tip özel erişimcileriyle birlikte, getStringValue(*fieldname*);

Bkz. “JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et” sayfa 815.

Kuyruk yöneticisi verileri dönüştürme

Bir JMS istemcisi programı bir ileti aldığında, WebSphere MQ V7.0' da kod sayfası dönüşümü kuyruk yöneticisi tarafından gerçekleştirilebilir. Dönüştürme, C programı için gerçekleştirilen dönüştürmenin aynısıdır. A C program sets MQGMO_CONVERT as an MQGET GetMsgOpts parameter option; see Şekil 131 sayfa 800. Bir kuyruk yöneticisi ileti alan bir JMS istemcisi programı için dönüştürme gerçekleştirir;

[5] Bir kural dışı durum: writeUTF kullanılarak yazılan veriler, 2 baytlık bir uzunluk alanıyla başlar

WMQ_RECEIVE_CONVERSION hedef özelliği WMQ_RECEIVE_CONVERSION_QMGRolarak ayarlandıysa, JMS istemcisi programı hedef özelliğini de ayarlayabilir; bkz. Şekil 130 sayfa 797.

V7.0öncesinde, JMS istemcisi dönüştürmeleri her zaman gerçekleştirirlerdi. JMS istemcisi veri dönüştürmesi, JMS istemcisi ile bilinen sayı ve tip ve uzunluk dizilerini dönüştürmek için kısıtlanmıştır. Veri yapılarını dönüştüremiyor; bkz. “JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et” sayfa 815. In V7.0, until fix pack 7.0.1.5, if the conversion can be performed by the queue manager, it is always performed by the queue manager. 7.0.1.5 düzeyinden itibaren, varsayılan dönüştürme davranışı V6.0ile aynı değere geri döner ve tüm dönüştürmeler JMS istemcisi tarafından gerçekleştirilir. From 7.0.1.5, or 7.0.1.4 with APAR IC72897, you can set a new destination option, WMQ_RECEIVE_CONVERSION, to control where conversion is performed, and WMQ_RECEIVE_CCSID, to set the target code page; see Şekil 130 sayfa 797.

```
((MQDestination)destination).setIntProperty(
              WMQConstants.WMQ_RECEIVE_CONVERSION,
              WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
```

Ya da,

```
((MQDestination)destination).setReceiveConversion
             (WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
```

Şekil 130. Kuyruk yöneticisi veri dönüştürmesini etkinleştir

Kuyruk yöneticisi dönüşümünün ana yararı, JMS dışı uygulamalarla ileti alışverişi yaparken gelir. İletideki Format alanı tanımlıysa ve hedef karakter kümesi ya da kodlama iletiyle farklıysa, kuyruk yöneticisi hedef uygulama için veri dönüştürme işlemi gerçekleştirir (uygulama bunu isterse). The queue manager converts message data formatted according to one of the predefined WebSphere MQ message types, such as a CICS bridge header (MQCIH). If the Format field is user-defined, the queue manager looks for a data conversion exit with the name provided in the Format field.

Kuyruk yöneticisi verileri dönüştürme, "alıcı iyi olur" tasarım örüntüleriyle en iyi şekilde kullanılır. Bir JMS istemcisinin dönüştürme gerçekleştirmesine gerek yoktur. JMS dışı giriş programı, iletinin gerekli kod sayfası ve kodlamada teslim edilmesini sağlamak için dönüştürme çıkışa dayanır. Bir gönderen JMS istemcisi ve JMS dışı alıcı ile, örnek IBM WebSphere MQ ön-ve post-V7.0için geçerlidir. IBM WebSphere MQ V7.0ile, dönüştürme çıkışı, alan bir JMS programı için de çağrılabilir.

Kuyruk yöneticisinin kendi kayıt biçimlenmiş verilerinizi dönüştürmesini sağlamak için, veri dönüştürme çıkış yardımcı programını (**crtmqcvx**) kullanarak bir veri dönüştürme çıkışı yaratabilirsiniz. Kendi kayıt biçiminizi oluşturabilir, bir Java sınıfı olarak erişmek için com.ibm.mq.headers ' u kullanabilir ve dönüştürmek için kendi dönüştürme çıkışınızı kullanabilirsiniz. z/OS üzerinde yardımcı program **CSQUCVX**olarak adlandırılır ve IBM i, **CVTMQMDTA**üzerinde. Bkz. “JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et” sayfa 815.

İleti kanalı veri dönüştürme

WebSphere MQ Sender, Server, Cluster-receiver, and Cluster-sender channels have a message conversion option, DöN. Bir iletinin içeriği, isteğe bağlı olarak bir ileti gönderildiğinde dönüştürülebilmektedir. Dönüştürme, kanalın gönderme sonunun sonunda gerçekleşir. Kümeli günlük nesnesi tanımlaması, karşılık gelen küme gönderici kanalını otomatik olarak tanımlamak için kullanılır.

İleti kanallarına göre veri dönüştürme genellikle diğer dönüştürme biçimlerini kullanmak olanaklı değilse kullanılır.

İleti dönüştürmesi için bir yaklaşım seçilmesi: "alıcı iyi olur"

Kod dönüştürmesi için WebSphere MQ uygulama tasarımında olağan yaklaşım "alıcı iyi olur"olur. "Günlük nesnesi iyi duruma gelir" , ileti dönüştürme sayısını azaltır. İleti aktarımı sırasında bazı aracı kuyruk yöneticisinde ileti dönüştürme işlemi başarısız olursa, beklenmeyen kanal hataları sorunu da önler. The

"alıcı iyi olur" rule is only broken if there is some reason why the receiver cannot make good. Alma platformu, örneğin doğru karakter takılmasına sahip olmayabilir.

"Günlük nesnesi iyi duruma getirir" , JMS istemcisi uygulamaları için de iyi bir genel rehberdir. Ancak belirli durumlarda, kaynakta ayarlanan doğru karakter kümesine dönüştürme daha verimli olabilir. Metin ya da sayısal tipler içeren bir ileti gönderildiğinde, JVM iç temsilinden dönüştürme gerçekleşmelidir. Alıcı bir JMS istemcisi değilse, alıcı tarafından gerekli olan karakter kümesine dönüştürme, JMS dışı alıcının dönüştürmeyi gerçekleştirmesi gerisini kaldırabilir. Alıcı bir JMS istemciyse, yine de, ileti verilerinin kodunu çözmek ve Java primitives ve objects yaratmak için yine de dönüştürülebiliyor.

JMS istemci uygulamaları ile C gibi bir dilde yazılmış uygulamalar arasındaki fark, Java 'nın veri dönüştürme gerçekleştirmesi gerekir. Bir Java uygulaması, numaraları ve metni iç gösterimlerinden, iletilerde kullanılan kodlanmış bir biçime dönüştürmelidir.

By setting destination, or message properties, you can set the character set and encoding used by WebSphere MQ to encode numbers and text in messages. Normally, you would leave the character set as `1208` and encoding as `Native`.

WebSphere MQ byte dizilerini dönüştürmez. Dizgileri ve karakter dizilerini bayt dizilerine kodlamak için `java.nio.charset` paketini kullanın. `Charset` , bir dizeyi ya da karakter dizisini bayt dizisine dönüştürmek için kullanılan karakter kümesini belirtir. You can also decode a byte array into a string or character array using a `Charset`. Dizgiler ve karakter dizileri kodlanırken `java.nio.charset.Charset.defaultCodePage` ' a güvenmek için iyi bir uygulama değildir. Varsayılan `Charset` genellikle Windowsüzerinde `windows-1252` ve UNIXüzerinde `UTF-8` ' dir. `windows-1252` , tek baytlık karakter takılıdır ve `UTF-8` çok baytlı bir karakter kümesidir.

Diğer JMS uygulamalarıyla ileti alışverişi yaparken, genellikle hedef karakter takımı ve kodlama özelliklerini varsayılan değer olarak `UTF-8` ve `Native` varsayılan değerlerinde bırakın. Bir JMS uygulamasıyla sayılar ya da metin içeren iletiler alışverişinde bulunsanız, amaçınıza uyan `JMSTextMessage`, `JMSStreamMessage`, `JMSMapMessage`ya da `JMSObjectMessage` ileti tiplerinden birini seçin. Yapılacak başka dönüştürme görevi yok.

Kayıt biçimi kullanan JMS dışı uygulamalarla ileti alışverişi yapıyorsanız, bu, kayıt biçimi daha karmaşıktır. Tüm kayıt metin içermiyorsa ve bir `JMSTextMessage`olarak aktarılamıyorsa, uygulamada metni kodlamanız ve kodu çözmeniz gerekir. Hedef ileti tipini `MQ`olarak ayarlayın ve JMS için IBM WebSphere MQ sınıflarının ek üstbilgi eklemesini ve ileti verisine bilgi etiketlenmesini önlemek için `JMSBytesMessage` kullanın. Sayı ve bayt yazmak için `JMSBytesMessage` yöntemlerini ve `Charset` sınıfı metni belirtik olarak bayt dizilerine dönüştürür. Bir dizi faktör, karakter takımı seçiminizi etkileyebilir:

- Performans: Metni, en fazla sayıda sunucu üzerinde kullanılan bir karakter kümesine dönüştürerek dönüştürmenin sayısını azaltabilir misiniz?
- Uniformity: Tüm iletileri aynı karakter kümesinde aktarın.
- Richness: Uygulamaların kullanması gereken tüm kod noktaları hangi karakter kümelerinde bulunur?
- Basitlik: Tek baytlık karakter kümelerinin kullanımı, değişken uzunluktan ve çok baytlı karakter kümelerinden daha basitdir.

Bkz. “JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et” sayfa 815. JMS dışı uygulamalarla değiş tokuş edilen iletilerin dönüştürülmesine ilişkin örnekler için.

Örnekler

İleti tipleri ve dönüştürme tipleri tablosu

Çizelge 119. İleti tipleri ve dönüştürme tipleri

	Dönüştürme tipi			
İleti tipi	**Metin**	**Sayısal**	**Diğer**	**Yok**
JMSObjectMessage				getObject setObject
JMSTextMessage	getText setText			
JMSBytesMessage	readUTF writeUTF	readDouble readFloat readInt readLong readShort readUnsignedShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readUnsignedByte readBytes readChar writeByte writeBytes writeChar
JMSStreamMessage	readString writeString	readDouble readFloat readInt readLong readShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readBytes readChar writeByte writeBytes writeChar
JMSMapMessage	getString setString	getDouble getFloat getInt getLong getShort setDouble setFloat setInt setLong setShort	getBoolean getObject setBoolean setObject	getByte getBytes readChar setByte setBytes setChar

```
gmo.Options = MQGMO_WAIT         /* wait for new messages           */
            | MQGMO_NO_SYNCPOINT /* no transaction                  */
            | MQGMO_CONVERT;     /* convert if necessary            */

     while (CompCode != MQCC_FAILED) {
     buflen = sizeof(buffer) - 1; /* buffer size available for GET   */
     memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
     memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
     md.Encoding       = MQENC_NATIVE;
     md.CodedCharSetId = MQCCSI_Q_MGR;

           MQGET(Hcon,          /* connection handle               */
           Hobj,                /* object handle                   */
           &md,                 /* message descriptor              */
           &gmo,                /* get message options             */
           buflen,              /* buffer length                   */
           buffer,              /* message buffer                  */
           &messlen,            /* message length                  */
           &CompCode,           /* completion code                 */
           &Reason);            /* reason code                     */
```

Şekil 131. Kod parçacığı: amqsget0.c

JMSBytesMessageiçine metin gönderme ve alma

Şekil 132 sayfa 800 içindeki kod, BytesMessageiçinde bir dizgi gönderir. Basitlik için, örnek, JMSTextMessage ' un daha uygun olduğu tek bir dizgi gönderir. To receive a text string in bytes message containing a mixture of types, you must know the length of the string in bytes, called *TEXT_LENGTH* in Şekil 133 sayfa 800. Sabit sayıda karakter içeren bir dizgi için bile, bayt gösteriminin uzunluğu daha uzun olabilir.

```
BytesMessage bytes = session.createBytesMessage();
String codePage = CCSID.getCodepage(((MQDestination) destination)
                  .getIntProperty(WMQConstants.WMQ_CCSID));
bytes.writeBytes("In the destination code page".getBytes(codePage));
producer.send(bytes);
```

Şekil 132. JMSBytesMessageiçine String gönderme

```
BytesMessage message = (BytesMessage)consumer.receive();
int TEXT_LENGTH = new Long(message.getBodyLength())).intValue();
byte[] textBytes = new byte[TEXT_LENGTH];
message.readBytes(textBytes, TEXT_LENGTH);
String codePage = message.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET);
String textString = new String(textBytes, codePage);
```

Şekil 133. JMSBytesMessage' dan bir String alma

İlgili kavramlar

JMS istemcisi ileti dönüştürme ve kodlama
JMS istemcisi ileti dönüştürme ve kodlamasını yapmak için kullandığınız yöntemler, her dönüştürme tipine ilişkin kod örnekleriyle listelenir.

Kuyruk yöneticisi verileri dönüştürme
Kuyruk yöneticisi verileri dönüştürme her zaman JMS istemcilerinden ileti alan JMS dışı uygulamalar için kullanılabilir. V7.0' dan bu yana, ileti alan JMS istemcileri de kuyruk yöneticisi veri dönüştürmesini kullanır. 7.0.1.5ya da APAR IC72897ile 7.0.1.4 , kuyruk yöneticisi veri dönüştürme isteğe bağlıdır.

İlgili görevler

JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et

Bir veri dönüştürme çıkışı tasarlamak ve oluşturmak için bu görevde önerilen adımları izleyin ve JMSBytesMessageeile JMS dışı bir uygulamayla iletileri değiş tokuş eden bir JMS istemcisi uygulaması. JMS dışı bir uygulamayla biçimlendirilmiş bir iletinin değiş tokası, veri dönüştürme çıkışı çağrılmadan ya da çağrılmadan gerçekleşebilir.

İlgili başvurular

JMS ileti tipleri ve dönüştürme
İleti tipi seçimi, ileti dönüştürmeye yaklaşımınızı etkiler. İleti dönüştürme ve ileti tipi etkileşimi, JMS ileti tipleri JMSObjectMessage, JMSTextMessage, JMSMapMessage, JMSStreamMessageve JMSBytesMessageJMS ileti tipleri için açıklanmıştır.

JMS ileti tipleri ve dönüştürme
İleti tipi seçimi, ileti dönüştürmeye yaklaşımınızı etkiler. İleti dönüştürme ve ileti tipi etkileşimi, JMS ileti tipleri JMSObjectMessage, JMSTextMessage, JMSMapMessage, JMSStreamMessageve JMSBytesMessageJMS ileti tipleri için açıklanmıştır.

JMSObjectMessage

JMSObjectMessage , bir nesne ve başvuruda bulunduğu tüm nesneleri, JVM tarafından bayt akışına diziselleştirilmiş bir nesne içerir. Metin, UTF-8' a serileştirilmiş ve 65534 bayttan daha fazla olmayan dizgiler ya da karakter dizileriyle sınırlandırılır. An advantage of JMSObjectMessage is that applications are not involved in any data conversion issues as long as they use only the methods and attributes of the object. JMSObjectMessage , uygulama programcısı olmayan karmaşık nesneler için, bir iletinin bir iletiye nasıl kodlacağını göz önünde bulundurarak karmaşık nesneler için veri dönüştürme olanağı sağlar. JMSObjectMessage kullanılmasının dezavantajı, yalnızca diğer JMS uygulamalarıyla değiş tokuş edilebilir. Diğer JMS ileti tiplerinden birini seçerek JMS iletileri JMS dışı uygulamalarla değiş tokuş etmek mümkündür.

“JMSObjectMessagegönderme ve alma” sayfa 803 , bir iletide değiştirilmekte olan bir String nesnesini gösterir.

Bir JMS istemcisi uygulaması, JMSObjectMessage ' yi yalnızca JMS stili gövdesi olan bir iletide alabilir. Hedef, bir JMS stil gövdesi belirtmelidir.

JMSTextMessage

JMSTextMessage , tek bir metin dizgisi içerir. When a text message is sent, the text Format is set to "MQSTR ", WMQConstants.MQFMT_STRING. Metnin CodedCharacterSetId değeri, hedefi için tanımlanan kodlanmış karakter takımı tanıtıcısı olarak ayarlanır. Metin, WebSphere MQtarafından CodedCharacterSetId içine kodlanır. CodedCharacterSetId ve Format alanları, ileti tanımlayıcısına (MQMD) ya da MQRFH2içindeki JMS alanlarına ayarlanır. İleti bir WMQ_MESSAGE_BODY_MQ ileti gövdesi stiline sahip olarak tanımlandıysa ya da gövde stili belirtilmediyse, ancak hedef hedef WMQ_TARGET_DEST_MQise, ileti tanımlayıcı alanları ayarlanır. Otherwise the message has a JMS RFH2 and the fields are set in the fixed part of the MQRFH2.

Bir uygulama, bir hedef için tanımlanan kodlanmış karakter takımı tanıtıcısını geçersiz kılabilir. JMS_IBM_XX_ENCODE_CASE_ONE charter_set ileti özelliğini kodlanmış bir karakter kümesi tanıtıcısı olarak ayarlamalıdır; örneğin, “JMSTextmessagegönderme ve alma” sayfa 803içindeki örneğe bakın.

JMS istemcisi consumer.receive yöntemi kuyruk yöneticisi dönüşümünü çağırdığında isteğe bağlıdır. Queue manager conversion is enabled by setting the destination property WMQ_RECEIVE_CONVERSION to WMQ_RECEIVE_CONVERSION_QMGR. Kuyruk yöneticisi, iletiyi JMS istemcisine aktarmadan önce ileti için belirtilen JMS_IBM_CHARACTER_SET ' dan metin iletisini dönüştürür. Hedef farklı bir WMQ_RECEIVE_CCSIDolmadıkça, dönüştürülen iletinin karakter kümesi 1208(UTF-8) olarak ayarlanır. JMSTextMessage ' a gönderme yapan iletide CodedCharacterSetId , hedef karakter kümesi tanıtıcısı olarak güncellenir. The text is decoded from the target character set into Unicode by the getText method; see the example in “JMSTextmessagegönderme ve alma” sayfa 803.

A JMSTextMessage can be sent in an MQ-style message body, without a JMS MQRFH2 header. Uygulama tarafından geçersiz kılınmadıkça, hedef özniteliklerin değeri WMQ_MESSAGE_BODY ve WMQ_TARGET_DEST ileti gövdesi biçemini belirler. The application can override the values set on the destination

by calling `destination.setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_MQ)` or `destination.setTargetClient(WMQConstants.WMQ_TARGET_DEST_MQ)`.

If you send a `JMSTextMessage` with an MQ style body by sending it to a destination with `WMQ_MESSAGE_BODY` set to `WMQ_MESSAGE_BODY_MQ`, you cannot receive it as a `JMSTextMessage` from the same destination. All messages received from a destination with `WMQ_MESSAGE_BODY` set to `WMQ_MESSAGE_BODY_MQ` are received as a `JMSBytesMessage`. İletiyi `JMSTextMessage` olarak almaya çalışırsanız, bir kural dışı durum oluşmasına neden olur: `ClassCastException: com.ibm.jms.JMSBytesMessage cannot be cast to javax.jms.TextMessage`.

Not: Bir `JMSBytesMessage` içindeki metin, JMS istemcisi tarafından dönüştürülmez. İstemci yalnızca iletideki metni bayt dizisi olarak alabilir. Kuyruk yöneticisi dönüşümü etkinleştirildiyse, metin kuyruk yöneticisi tarafından dönüştürülür, ancak JMS istemcisi bunu yine de `JMSBytesMessage`eiçinde bayt dizisi olarak almalıdır.

Bir `JMSTextMessage` değerinin MQ ya da JMS gövdesi stiliyle gönderilip gönderilmediğini denetlemek için `WMQ_TARGET_DEST` özelliğinin kullanılması genellikle daha iyi olur. Daha sonra, iletiyi `WMQ_TARGET_DEST` değerine ayarlanmış bir hedeften `WMQ_TARGET_DEST_MQ` ya da `WMQ_TARGET_DEST_JMS`olarak ayarlayabilirsiniz. `WMQ_TARGET_DEST` , alıcı üzerinde hiçbir etkiye sahip değildir.

JMSMapMessage ve JMSStreamMessage

Bu iki JMS ileti tipi benzerdir. You can read and write primitive types to the messages using methods based on the `DataInputStream` and `DataOutputStream` interfaces; see “İleti tipleri ve dönüştürme tipleri tablosu” sayfa 806. Ayrıntılar “JMS istemcisi ileti dönüştürme ve kodlama” sayfa 807içinde açıklanmıştır. Her temel öğe etiketlenmiş; bkz. “JMS ileti gövdesi” sayfa 792.

Sayısal veriler okunuyor ve XML metni olarak kodlanmış iletiye yazıldı. Hedef özelliğe hiçbir başvuru yapılmadı, `JMS_IBM_ENCODING`. Metin verileri, `JMSTextMessage`eiçindeki metinle aynı şekilde işlem görür. Örneğin, Şekil 138 sayfa 804içindeki örnek tarafından oluşturulan ileti içeriğine bakmanız gerekiyorsa, tüm ileti verileri, 37 karakter kümesi değeriyle gönderildiği için EBCDIC ' de olur.

Bir `JMSMapMessage` ya da `JMSStreamMessage`eiçine birden çok öğe gönderebilirsiniz.

Verilerin tek tek öğelerini bir `JMSMapMessage`'den ya da bir `JMSStreamMessage`' den konumuna göre alabilirsiniz. İletide saklanan `CodedCharacterSetId` değeri kullanılarak bir alma ya da okuma yöntemi çağrıldığında, her öğenin kodu çözülür. Öğeyi almak için kullanılan yöntem, gönderilen tip için farklı bir tip döndürürse, tip dönüştürülür. Tip dönüştürülemezse, kural dışı durum oluşur. Ayrıntılar için Sınıf JMSStreamMessage başlıklı konuya bakın. “Sending data in a JMSStreamMessage and JMSMapMessage” sayfa 804 içindeki örnek, tip dönüşümünü gösterir ve `JMSMapMessage` içeriğini sıradan dışarı çıkardır.

`JMSMapMessage` ve `JMSStreamMessage` için `MQRFH2.format` alanı, "MQSTR "olarak ayarlıdır. If the destination property `WMQ_RECEIVE_CONVERSION` is set to `WMQ_RECEIVE_CONVERSION_QMGR`, the message data is converted by the queue manager before being sent to the JMS client. İletinin `MQRFH2.CodedCharacterSetId` değeri, hedefin `WMQ_RECEIVE_CCSID` ' idir. The `MQRFH2.Encoding` is `Native`. If `WMQ_RECEIVE_CONVERSION` is `WMQ_RECEIVE_CONVERSION_CLIENT_MSG` the `CodedCharacterSetId` and `Encoding` of the `MQRFH2` is the value set by the sender.

Bir JMS istemcisi uygulaması, yalnızca JMS stili gövdesi olan ve MQ stilinde bir gövde belirtmeyen bir hedeften `JMSMapMessage` ya da `JMSStreamMessage` iletisini alabilir.

JMSBytesMessage

Bir `JMSBytesMessage` birden çok ilkel tip içerebilir. You can read and write primitive types to the messages using methods based on the `DataInputStream` and `DataOutputStream` interfaces; see “İleti tipleri ve dönüştürme tipleri tablosu” sayfa 806. Ayrıntılar “JMS ileti tipleri ve dönüştürme” sayfa 801içinde açıklanmıştır.

The encoding of numeric data in the message is controlled by the value of `JMS_IBM_ENCODING` that is set before writing numeric data to the `JMSBytesMessage`. Bir uygulama, `JMS_IBM_ENCODING`ileti özelliğini ayarlayarak `JMSBytesMessage` için tanımlanan varsayılan `Native` kodlamasını geçersiz kılabilir.

Text data can be read and written in UTF-8 using the `readUTF` and `writeUTF`, or in Unicode using the `readChar` and `writeChar` methods. `CodedCharacterSetId`' yi kullanan hiçbir yöntem yok. Diğer bir seçenek olarak, JMS istemcisi, `Charset` sınıfını kullanarak metni bayt olarak kodlayabilir ve byte olarak kodlayabilir. JMS için WebSphere MQ sınıfları dönüştürülmeden JVM ve ileti arasındaki baytları aktarır; bkz. "JMSBytesMessageiçine metin gönderme ve alma" sayfa 804.

MQ uygulamasına gönderilen bir `JMSBytesMessage` , genellikle JMS `MQRFH2` üstbilgisi olmadan bir MQstili ileti gövdesinde gönderilir. Bir JMS uygulamasına gönderilirse, ileti gövdesi stili genellikle JMS ' dir. Uygulama tarafından geçersiz kılınmadıkça, hedef özniteliklerin değeri `WMQ_MESSAGE_BODY` ve `WMQ_TARGET_DEST` ileti gövdesi biçemini belirler. The application can override the values set on the destination by calling `destination.setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_MQ)` or `destination.setTargetClient(WMQConstants.WMQ_TARGET_DEST_MQ)`.

Bir MQ stil gövdesi ile bir `JMSBytesMessage` gönderdiğinizde, iletiyi bir MQ ya da JMS iletisi gövdesi stilini tanımlayan bir hedeften alabilirsiniz. JMS stili gövdesi içeren bir `JMSBytesMessage` gönderdiğinizde, iletiyi JMS ileti gövdesi stilini tanımlayan bir hedeften almanız gerekir. Bunu yapmazsanız, `MQRFH2` , kullanıcı ileti verilerinin bir parçası olarak kabul edilir; bu, beklediğiniz gibi olmayabilir.

İletinin MQ ya da JMS gövdesi stiline sahip olup olmadığı, `WMQ_TARGET_DEST`Tayarından etkilenmez.

İleti daha sonra, kuyruk yöneticisi tarafından, ileti verileri için bir `Format` sağlandıysa ve kuyruk yöneticisi veri dönüştürme etkinleştirilmişse, ileti daha sonra dönüştürülebilirdi. İleti verilerinin biçimini belirtmekten başka bir şey için biçim alanını kullanmayın ya da boş bırakın, `MQConstants.MQFMT_NONE`

Bir `JMSBytesMessage`' ta birden çok öğe gönderebilirsiniz. İleti için tanımlanan kodlamayı kullanarak ileti gönderildiğinde, her sayısal öğe dönüştürülür.

You can retrieve the individual items of data from `JMSBytesMessage`. İletiyi yaratmak için yazma yöntemleriyle aynı sırada okuma yöntemlerini çağırın. İletide saklanan `Encoding` değeri kullanılarak ileti çağrıldığında, her sayısal öğe dönüştürülür.

`JMSMapMessage` ve `JMSStreamMessage`' ten farklı olarak, `JMSBytesMessage` yalnızca uygulama tarafından yazılan verileri içerir. Bir `JMSMapMessage` ve `JMSStreamMessage`içindeki öğeleri tanımlamak için kullanılan XML etiketleri gibi ileti verilerinde ek veri saklanmaz. Bu nedenle, diğer uygulamalar için biçimlenmiş iletileri aktarmak için `JMSBytesMessage` simgesini kullanın.

Converting between `JMSBytesMessage` and `DataInputStream` and `DataOutputStream` is useful in some applications. Code based on the example, "DataInputStream ve DataOutputStreamkullanarak ileti okuma ve yazma" sayfa 805, is necessary to use the `com.ibm.mq.header` package with JMS.

Örnekler

`JMSObjectMessage`gönderme ve alma

```
ObjectMessage omo = session.createObjectMessage();
omo.setObject(new String("A string"));
producer.send(omo);
...
ObjectMessage omi = (ObjectMessage)consumer.receive();
System.out.println((String)omi.getObject());
...
A string
```

Şekil 134. `JMSObjectMessage`gönderme ve alma

`JMSTextmessage`gönderme ve alma

Metin iletisi, farklı karakter kümelerinde metin içeremez. Örnekte, farklı karakter kümelerindeki metin, iki farklı ileti halinde gönderilir.

```
TextMessage tmo = session.createTextMessage();
tmo.setText("Sent in the character set defined for the destination");
producer.send(tmo);
```

Şekil 135. Metin iletisini hedef tarafından tanımlanan karakter kümesiyle gönder

```
TextMessage tmo = session.createTextMessage();
tmo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 37);
tmo.setText("Sent in EBCDIC character set 37");
producer.send(tmo);
```

Şekil 136. ccsid 37 'de metin iletisi gönder

```
TextMessage tmi = (TextMessage)consumer.receive();
System.out.println(tmi.getText());
...
Sent in the character set defined for the destination
```

Şekil 137. Metin iletisi al

Sending data in a JMSStreamMessage and JMSMapMessage

```
StreamMessage smo = session.createStreamMessage();
smo.writeString("256");
smo.writeInt(512);
smo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 37);
producer.send(smo);
...
MapMessage mmo = session.createMapMessage();
mmo.setString("First", "256");
mmo.setInt("Second", 512);
mmo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 37);
producer.send(mmo);
...
StreamMessage smi = (StreamMessage)consumer.receive();
System.out.println("Stream: First as float "   + smi.readFloat() +
                           " Second as String " + smi.readString());
...
Stream: First as float: 256.0, Second as String: 512
...
MapMessage mmi = (MapMessage)consumer.receive();
System.out.println("Map: Second as String " + mmi.getString("Second") +
                       " First as double "  + mmi.getDouble("First"));
...
Map: Second as String: 512, First as double: 256.0
```

Şekil 138. Send data in JMSStreamMessage and JMSMapMessage

JMSBytesMessageiçine metin gönderme ve alma

Şekil 139 sayfa 805 içindeki kod, BytesMessageiçinde bir dizgi gönderir. Basitlik için, örnek, JMSTextMessage ' un daha uygun olduğu tek bir dizgi gönderir. To receive a text string in bytes message containing a mixture of types, you must know the length of the string in bytes, called *TEXT_LENGTH* in Şekil 140 sayfa 805. Sabit sayıda karakter içeren bir dizgi için bile, bayt gösteriminin uzunluğu daha uzun olabilir.

```
BytesMessage bytes = session.createBytesMessage();
String codePage = CCSID.getCodepage(((MQDestination) destination)
                  .getIntProperty(WMQConstants.WMQ_CCSID));
bytes.writeBytes("In the destination code page".getBytes(codePage));
producer.send(bytes);
```

Şekil 139. JMSBytesMessageiçine String gönderme

```
BytesMessage message = (BytesMessage)consumer.receive();
int TEXT_LENGTH = new Long(message.getBodyLength())).intValue();
byte[] textBytes = new byte[TEXT_LENGTH];
message.readBytes(textBytes, TEXT_LENGTH);
String codePage = message.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET);
String textString = new String(textBytes, codePage);
```

Şekil 140. JMSBytesMessage' dan bir String alma

DataInputStream ve DataOutputStreamkullanarak ileti okuma ve yazma

Şekil 141 sayfa 805 içindeki kod, DataOutputStreamkullanarak bir JMSBytesMessage oluşturur.

```
ByteArrayOutputStream bout = new ByteArrayOutputStream();
DataOutputStream dout = new DataOutputStream(bout);
BytesMessage messageOut = prod.session.createBytesMessage();
// messageOut.setIntProperty(WMQConstants.JMS_IBM_ENCODING,
//                           ((MQDestination) (prod.destination)).getIntProperty
//                           (WMQConstants.WMQ_ENCODING));
int ccsidOut = (((MQDestination)prod.destination).getIntProperty(WMQConstants.WMQ_CCSID));
String codePageOut = CCSID.getCodepage(ccsidOut);
dout.writeInt(ccsidOut);
dout.write(codePageOut.getBytes(codePageOut));
messageOut.writeBytes(bout.toByteArray());
producer.send(messageOut);
```

Şekil 141. DataOutputStreamkullanarak bir JMSBytesMessage gönderin

JMS_IBM_ENCODING özelliğini belirleyen deyim açıklama satırı olarak geçersiz kılınmaktadır. Bu deyim, doğrudan bir JMSBytesMessage'a yazıyorsa, ancak DataOutputStream' a yazarken hiçbir etkisi yoktur. DataOutputStream ' a yazılan sayılar Native kodlamasında kodlanır. JMS_IBM_ENCODING ayarının bir etkisi yok.

Şekil 142 sayfa 805 içindeki kod, DataInputStreamkullanan bir JMSBytesMessage alır.

```
static final int ccsidIn_SIZE = (Integer.SIZE)/8;
...
connection.start();
BytesMessage messageIn = (BytesMessage) consumer.receive();
int messageLength = new Long(messageIn.getBodyLength()).intValue();
byte [] bin = new byte[messageLength];
messageIn.readBytes(bin, messageLength);
DataInputStream din = new DataInputStream(new ByteArrayInputStream(bin));
int ccsidIn = din.readInt();
byte [] codePageByte = new byte[messageLength - ccsidIn_SIZE];
din.read(codePageByte, 0, codePageByte.length);
System.out.println("CCSID " + ccsidIn + " code page " + new String(codePageByte,
                    messageIn.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET)));
```

Şekil 142. DataInputStreamkullanarak bir JMSBytesMessage alma

Kod sayfası, giriş iletisi verilerinin kod sayfası özelliği (JMS_IBM_CHARACTER_SET) kullanılarak yazdırılır. On input JMS_IBM_CHARACTER_SET is a Java code page and not a numeric coded character set identifier.

İleti tipleri ve dönüştürme tipleri tablosu

Çizelge 120. İleti tipleri ve dönüştürme tipleri

	Dönüştürme tipi			
İleti tipi	**Metin**	**Sayısal**	**Diğer**	**Yok**
JMSObjectMessage				getObject setObject
JMSTextMessage	getText setText			
JMSBytesMessage	readUTF writeUTF	readDouble readFloat readInt readLong readShort readUnsignedShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readUnsignedByte readBytes readChar writeByte writeBytes writeChar
JMSStreamMessage	readString writeString	readDouble readFloat readInt readLong readShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readBytes readChar writeByte writeBytes writeChar
JMSMapMessage	getString setString	getDouble getFloat getInt getLong getShort setDouble setFloat setInt setLong setShort	getBoolean getObject setBoolean setObject	getByte getBytes readChar setByte setBytes setChar

İlgili kavramlar

JMS ileti dönüştürme yaklaşımları

Bir dizi veri dönüştürme yaklaşımları JMS uygulama tasarımcılarına açıktır. Bu yaklaşımlar münhasır değildir; bazı uygulamalar bu yaklaşımların bir birleşimini kullanma olasılığının yüksek olduğu bir tür. Uygulamanız yalnızca metin alışverişi yapmışsa ya da yalnızca diğer JMS uygulamalarıyla ileti alışverişi yapmışsa, normalde veri dönüştürmeyi dikkate almayın. Veri dönüştürme işlemi sizin için otomatik olarak WebSphere MQtarafından gerçekleştirilir.

JMS istemcisi ileti dönüştürme ve kodlama
JMS istemcisi ileti dönüştürme ve kodlamasını yapmak için kullandığınız yöntemler, her dönüştürme tipine ilişkin kod örnekleriyle listelenir.

Kuyruk yöneticisi verileri dönüştürme
Kuyruk yöneticisi verileri dönüştürme her zaman JMS istemcilerinden ileti alan JMS dışı uygulamalar için kullanılabilir. V7.0' dan bu yana, ileti alan JMS istemcileri de kuyruk yöneticisi veri dönüştürmesini kullanır. 7.0.1.5ya da APAR IC72897ile 7.0.1.4 , kuyruk yöneticisi veri dönüştürme isteğe bağlıdır.

İlgili görevler

JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et
Bir veri dönüştürme çıkışı tasarlamak ve oluşturmak için bu görevde önerilen adımları izleyin ve `JMSBytesMessage`ile JMS dışı bir uygulamayla iletileri değiş tokuş eden bir JMS istemcisi uygulaması. JMS dışı bir uygulamayla biçimlendirilmiş bir iletinin değiş tokası, veri dönüştürme çıkışı çağrılmadan ya da çağrılmadan gerçekleşebilir.

JMS istemcisi ileti dönüştürme ve kodlama
JMS istemcisi ileti dönüştürme ve kodlamasını yapmak için kullandığınız yöntemler, her dönüştürme tipine ilişkin kod örnekleriyle listelenir.

Dönüştürme ve kodlama, Java temel öğeleri ya da nesneler JMS iletilerine okunurken ya da JMS iletilerine yazıldığında oluşur. Dönüştürme, kuyruk yöneticisi veri dönüştürme ve uygulama verileri dönüşümünden ayırt etmek için JMS istemcisi veri dönüştürmesi olarak adlandırılır. Dönüştürme işlemi, verilerin bir JMS iletisinden okunduğu ya da bir JMS iletisine yazıldığı durumlarda gerçekleşir. Metin, iç 16 bit Unicode gösterimine ya da iç 16 bit 'e dönüştürülür.[6]İletilerde metin için kullanılan karakter kümesine. Sayısal veriler, ileti için tanımlanan kodlamaya dönüştürülmüş ve Java temel sayısal tiplerinden kodlanır. Dönüştürmenin gerçekleştirilip gerçekleştirilmeyeceğini ve hangi dönüştürme tipinin gerçekleştirileceğini, JMS ileti tipine ve okuma ya da yazma işlemine bağlıdır.

Çizelge 121 sayfa 807 , gerçekleştirilen dönüştürme tipine göre farklı JMS ileti tipleri için okuma ve yazma yöntemlerini kategorilere ayırır. Dönüştürme tipleri, çizelgenin ardından gelen metinde açıklanır.

Çizelge 121. İleti tipleri ve dönüştürme tipleri

	Dönüştürme tipi			
İleti tipi	**Metin**	**Sayısal**	**Diğer**	**Yok**
`JMSObjectMessage`				`getObject` `setObject`
`JMSTextMessage`	`getText` `setText`			

[6] Bazı Unicode gösterimi 16 bitten fazlasını gerektirir. Java SE başvurularına bakın.

Çizelge 121. İleti tipleri ve dönüştürme tipleri (devamı var)				
	Dönüştürme tipi			
İleti tipi	**Metin**	**Sayısal**	**Diğer**	**Yok**
JMSBytesMessage	readUTF writeUTF	readDouble readFloat readInt readLong readShort readUnsignedShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readUnsignedByte readBytes readChar writeByte writeBytes writeChar
JMSStreamMessage	readString writeString	readDouble readFloat readInt readLong readShort writeDouble writeFloat writeInt writeLong writeShort	readBoolean readObject writeBoolean writeObject	readByte readBytes readChar writeByte writeBytes writeChar
JMSMapMessage	getString setString	getDouble getFloat getInt getLong getShort setDouble setFloat setInt setLong setShort	getBoolean getObject setBoolean setObject	getByte getBytes readChar setByte setBytes setChar

Metin

Hedef için varsayılan CodedCharacterSetId , 1208, UTF-8' dır. Varsayılan olarak, metin Unicode 'dan dönüştürülür ve bir UTF-8 metin dizgisi olarak gönderilir. Alma işlemi sırasında metin, istemci tarafından alınan iletteki kodlanmış karakter kümesinden Unicode 'a dönüştürülür.

setText ve writeString yöntemleri, metni Unicode 'dan hedef için tanımlanan karakter kümesinden dönüştürür. Bir uygulama, JMS_IBM_CHAREACTER_SETileti özelliğini ayarlayarak hedef karakter kümesini geçersiz kılabilir. JMS_IBM_XX_ENCODE_CASE_ONE charter_set, bir ileti gönderirken, sayısal kodlanmış karakter takımı tanıtıcısı olmalıdır[7].

"JMSTextmessagegönderme ve alma" sayfa 811 içindeki kod parçacıklar iki ileti gönderir. Biri hedef için tanımlanan karakter kümesinde, diğeri de uygulama tarafından tanımlanan 37 karakter kümesiyle gönderilir.

[7] JMS_IBM_CHARACTER_SET iletisi alınırken, Java Charset kod sayfası adı yer almaktadır.

getText ve readString yöntemleri, iletteki metni, iletide tanımlı karakter kümesinden Unicode 'a dönüştürür. The methods use the code page defined in the message property, JMS_IBM_XX_ENCODE_CASE_ONE charter_set. İleti, bir MQtipi iletiyse ve MQRFH2olmadığında, kod sayfası MQRFH2.CodedCharacterSetId ' tan eşlenir. İleti bir MQtipi iletiyse, MQRFH2değeri yoksa, kod sayfası MQMD.CodedCharacterSetId' tan eşlenir.

Şekil 147 sayfa 811 içindeki kod parçacığı, hedefe gönderilen iletiyi alır. İletideki metin IBM037 kod sayfasından Unicode 'a geri dönüştürüldü.

Not: Metnin, kodlanmış karakter takımı 37 'ye dönüştürülmesini denetmenin basit bir yolu, WebSphere MQ Explorer 'ı kullanmandır. Kuyruğa göz atın ve ileti alınmadan önce iletinin özelliklerini gösterin.

Contrast the code snippet in Şekil 146 sayfa 811 with the incorrect code snippet in Şekil 143 sayfa 809. Yanlış kod parçacısında metin dizgisi, uygulama tarafından iki kez, bir kez de WebSphere MQtarafından dönüştürülür.

```
TextMessage tmo = session.createTextMessage();
tmo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 37);
tmo.setText(new String("Sent in EBCDIC character set 37".getBytes(CCSID.getCodepage(37))));
producer.send(tmo);
```

Şekil 143. Kod sayfası dönüşümü yanlış

writeUTF yöntemi, metni Unicode 'dan 1208 'e, UTF-8' a dönüştürür. Metin dizilimi, 2 baytlık bir uzunlukla önceden ortaya çıktı. Metin dizilimin uzunluk üst sınırı 65534 byte 'tır. readUTF yöntemi, writeUTF yöntemi tarafından yazılan bir iletide bir öğeyi okur. Bu, writeUTF yöntemi tarafından yazılan bayt sayısını tam olarak okur.

Sayısal

Bir hedefe ilişkin varsayılan sayısal kodlama: Native. Java için Native kodlama sabiti, tüm platformlar için aynı olan 273, x'00000111' değerine sahiptir. Alma işlemi sırasında, iletteki sayılar sayısal Java prizlerine doğru biçimde dönüştürülür. Dönüştürme, iletide tanımlı olan kodlamayı ve okuma yönteminin döndürdüğü tipi kullanır.

The send method converts numbers that are added to a message by the set and write into the numeric encoding defined for the destination. Hedef kodlama, bir uygulamanın ileti özelliğini ayarlayarak bir ileti için geçersiz kılınabilir; örneğin, JMS_IBM_ENCODING; örneğin:

```
message.setIntProperty(WMQConstants.JMS_IBM_ENCODING,
WMQConstants.WMQ_ENCODING_INTEGER_REVERSED);
```

get ve read sayısal yöntemleri, iletteki sayıları, iletide tanımlanan sayısal kodlamadaki değerleri dönüştürür. Sayıları, read ya da get yöntemi tarafından belirtilen tipe dönüştürüyorlar; bkz. ENCODING özelliği . Yöntemler, JMS_IBM_ENCODINGiçinde tanımlanan kodlamayı kullanır. Bu kodlama MQRFH2.Encoding 'den eşlenir; ancak, ileti bir MQ-type iletisi değilse ve MQRFH2' den (.) sahip değildir. İleti bir MQtipi iletiyse, MQRFH2değeri yoksa, yöntemler MQMD.Encodingiçinde tanımlanan kodlamayı kullanır.

Şekil 148 sayfa 811 içindeki örnek, bir uygulamanın hedef biçimdeki bir sayıyı kodlamasını ve bir JMSStreamMessage' a göndermesini gösterir. Şekil 148 sayfa 811 içindeki örneği, Şekil 149 sayfa 812içindeki örneğe göre karşılaştırın. Fark, JMS_IBM_ENCODING ' un JMSBytesMessageolarak ayarlanması gerekir.

Not: Sayının doğru olarak kodlandığından emin olmak için basit bir yol, WebSphere MQ Explorer tarayıcısını kullanmanın bir yoludur. Kuyruğa göz atın ve tüketmeden önce iletinin özelliklerini gösterin.

Diğer

The boolean methods encode true and false as x'01' and x'00' in a JMSByteMessage, JMSStreamMessage, and JMSMapMessage.

UTF yöntemleri, Unicode 'u UTF-8 metin dizgilerine kodlayıp kodu çözer. Dizilimler, 65536 karakterden kısa ve 2 byte 'lık uzunluk alanlarından önce gelir.

Nesne yöntemleri, ilkel tipleri nesne olarak tamamlar. Sayısal ve metin tipleri, temel tipler, sayısal ve metin yöntemleri kullanılarak okunmuyorsa ya da yazılmışsa, kodlanır ya da dönüştürülür.

Yok

`readByte`, `readBytes`, `readUnsignedByte`, `writeByte`ve `writeBytes` yöntemleri, uygulama ile ileti arasında dönüştürme yapılmadan tek baytlar ya da bayt dizilerini alır ya da kokardır. `readChar` ve `writeChar` yöntemleri, uygulama ile ileti arasında dönüştürme yapılmadan 2 byte Unicode karakter alır ve yerleştirir.

Uygulama, `readBytes` ve `writeBytes` yöntemlerini kullanarak “JMSBytesMessageiçine metin gönderme ve alma” sayfa 812içinde olduğu gibi kendi kod noktası dönüştürmesini gerçekleştirebilir.

WebSphere MQ does not perform any code page conversion in the client as the message is a `JMSBytesMessage`, and because the `readBytes` and `writeBytes` methods are used. Bununla birlikte, bayt metni gösteriyorsa, uygulama tarafından kullanılan kod sayfasının, hedefin kodlanmış karakter takısıyla eşleştiğinden emin olun. İleti, kuyruk yöneticisi dönüştürme çıkışı tarafından yeniden dönüştürülebilirdi. Başka bir olasılık da, alan JMS istemci programının iletide metni temsil eden byte dizilerini, iletide `JMS_IBM_CHARACTER_SET` özelliğini kullanarak dizelere ya da karakterlere dönüştürme kuralını izleyebileceğidir.

Bu örnekte istemci, dönüştürmesi için hedef kodlanmış karakter kümesini kullanır:

```
bytes.writeBytes("In the destination code page".getBytes(
      CCSID.getCodepage(((MQDestination) destination)
           .getIntProperty(WMQConstants.WMQ_CCSID))));
```

Diğer bir seçenek olarak, istemci bir kod sayfası seçmiş olabilir ve sonra iletinin `JMS_IBM_XX_ENCODE_CASE_ONE charcter_set` özelliğinde karşılık gelen kodlanmış karakter kümesini ayarlayabilir. Java için WebSphere MQ sınıfları `JMS_IBM_XX_ENCODE_CASE_ONE charter_set` , `MQRFH2`içindeki JMS özelliklerinde ya da ileti tanımlayıcısında `CodedCharacterSetId` alanını ayarlamak için: `MQMD`:

```
String codePage = CCSID.getCodepage(37);
message.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, codePage);[8]
```

If a byte array is written into a `JMSStringMessage` or `JMSMapMessage`, WebSphere MQ classes for JMS does not perform data conversion, as the bytes are typed as hexadecimal data not as text in the `JMSStringMessage` and `JMSMapMessage`.

Uygulamanınızdaki karakterler byte 'ları gösteriyorsa, okunabilmek ve iletiye yazmak için kod noktalarını dikkate almalısınız. Şekil 144 sayfa 810 içindeki kod, hedef kodlanmış karakter kümesini kullanma kuralından sonra gelir. JVM ' yi JVM için varsayılan karakter kümesini kullanarak oluşturursanız, bayt içeriği platforma bağlıdır. A JVM on Pencereler typically has a default `Charset` of `windows-1252`, and UNIX, `UTF-8`. Windows ile UNIX arasında, metin alışverişi için bayt olarak belirtik bir kod sayfası seçmenizi gerektirir.

```
StreamMessage smo = producer.session.createStreamMessage();
smo.writeBytes("123".getBytes(CCSID.getCodepage(((MQDestination) destination)
           .getIntProperty(WMQConstants.WMQ_CCSID))));
```

Şekil 144. Hedef karakter takımı kullanılarak JMS`StreamMessage` içindeki bir dizeyi gösteren byte yazılması

---

[8] `SetStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET, codePage) currently accepts only numeric character set identifiers.`

Örnekler

JMSTextmessagegönderme ve alma

Metin iletisi, farklı karakter kümelerinde metin içeremez. Örnekte, farklı karakter kümelerindeki metin, iki farklı ileti halinde gönderilir.

```
TextMessage tmo = session.createTextMessage();
tmo.setText("Sent in the character set defined for the destination");
producer.send(tmo);
```

Şekil 145. Metin iletisini hedef tarafından tanımlanan karakter kümesiyle gönder

```
TextMessage tmo = session.createTextMessage();
tmo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 37);
tmo.setText("Sent in EBCDIC character set 37");
producer.send(tmo);
```

Şekil 146. ccsid 37 'de metin iletisi gönder

```
TextMessage tmi = (TextMessage)consumer.receive();
System.out.println(tmi.getText());
...
Sent in the character set defined for the destination
```

Şekil 147. Metin iletisi al

Kodlama örnekleri

Kodlamada gönderilmekte olan bir sayıyı gösteren örnekler, bir hedefe ilişkin tanımlar. Bir JMSBytesMessage ' nin JMS_IBM_ENCODING özelliğini, hedef için belirtilen değere ayarlamamış olmanız gerektiğini fark edin.

```
StreamMessage smo = session.createStreamMessage();
smo.writeInt(256);
producer.send(smo);
...
StreamMessage smi = (StreamMessage)consumer.receive();
System.out.println(smi.readInt());
...
256
```

Şekil 148. JMSStreamMessageiçindeki hedef kodlamayı kullanarak bir sayı gönderme

```
BytesMessage bmo = session.createBytesMessage();
bmo.writeInt(256);
int encoding = ((MQDestination) (destination)).getIntProperty
    (WMQConstants.WMQ_ENCODING)
bmo.setIntProperty(WMQConstants.JMS_IBM_ENCODING, encoding);
producer.send(bmo);
...
BytesMessage bmi = (BytesMessage)consumer.receive();
System.out.println(bmi.readInt());
...
256
```

Şekil 149. JMSBytesMessageiçindeki hedef kodlamayı kullanarak bir sayı gönderme

JMSBytesMessageiçine metin gönderme ve alma

Şekil 150 sayfa 812 içindeki kod, BytesMessageiçinde bir dizgi gönderir. Basitlik için, örnek, JMSTextMessage ' un daha uygun olduğu tek bir dizgi gönderir. To receive a text string in bytes message containing a mixture of types, you must know the length of the string in bytes, called *TEXT_LENGTH* in Şekil 151 sayfa 812. Sabit sayıda karakter içeren bir dizgi için bile, bayt gösteriminin uzunluğu daha uzun olabilir.

```
BytesMessage bytes = session.createBytesMessage();
String codePage = CCSID.getCodepage(((MQDestination) destination)
                  .getIntProperty(WMQConstants.WMQ_CCSID));
bytes.writeBytes("In the destination code page".getBytes(codePage));
producer.send(bytes);
```

Şekil 150. JMSBytesMessageiçine String gönderme

```
BytesMessage message = (BytesMessage)consumer.receive();
int TEXT_LENGTH = new Long(message.getBodyLength())).intValue();
byte[] textBytes = new byte[TEXT_LENGTH];
message.readBytes(textBytes, TEXT_LENGTH);
String codePage = message.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET);
String textString = new String(textBytes, codePage);
```

Şekil 151. JMSBytesMessage' dan bir String alma

İlgili kavramlar

JMS ileti dönüştürme yaklaşımları
Bir dizi veri dönüştürme yaklaşımları JMS uygulama tasarımcılarına açıktır. Bu yaklaşımlar münhasır değildir; bazı uygulamalar bu yaklaşımların bir birleşimini kullanma olasılığının yüksek olduğu bir tür. Uygulamanız yalnızca metin alışverişi yapmışsa ya da yalnızca diğer JMS uygulamalarıyla ileti alışverişi yapmışsa, normalde veri dönüştürmeyi dikkate almayın. Veri dönüştürme işlemi sizin için otomatik olarak WebSphere MQtarafından gerçekleştirilir.

Kuyruk yöneticisi verileri dönüştürme
Kuyruk yöneticisi verileri dönüştürme her zaman JMS istemcilerinden ileti alan JMS dışı uygulamalar için kullanılabilir. V7.0' dan bu yana, ileti alan JMS istemcileri de kuyruk yöneticisi veri dönüştürmesini kullanır. 7.0.1.5ya da APAR IC72897ile 7.0.1.4 , kuyruk yöneticisi veri dönüştürme isteğe bağlıdır.

İlgili görevler

JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et
Bir veri dönüştürme çıkışı tasarlamak ve oluşturmak için bu görevde önerilen adımları izleyin ve JMSBytesMessageile JMS dışı bir uygulamayla iletileri değiş tokuş eden bir JMS istemcisi uygulaması. JMS dışı bir uygulamayla biçimlendirilmiş bir iletinin değiş tokası, veri dönüştürme çıkışı çağrılmadan ya da çağrılmadan gerçekleşebilir.

İlgili başvurular

JMS ileti tipleri ve dönüştürme

İleti tipi seçimi, ileti dönüştürmeye yaklaşımınızı etkiler. İleti dönüştürme ve ileti tipi etkileşimi, JMS ileti tipleri `JMSObjectMessage`, `JMSTextMessage`, `JMSMapMessage`, `JMSStreamMessage`eve `JMSBytesMessage`JMS ileti tipleri için açıklanmıştır.

Kuyruk yöneticisi verileri dönüştürme

Kuyruk yöneticisi verileri dönüştürme her zaman JMS istemcilerinden ileti alan JMS dışı uygulamalar için kullanılabilir. V7.0' dan bu yana, ileti alan JMS istemcileri de kuyruk yöneticisi veri dönüştürmesini kullanır. 7.0.1.5ya da APAR IC72897ile 7.0.1.4 , kuyruk yöneticisi veri dönüştürme isteğe bağlıdır.

Kuyruk yöneticisi, ileti verileri için `CodedCharacterSetId`, `Encoding`ve `Format` değerlerini kullanarak ileti verilerinde karakter ve sayısal veriler dönüştürebilir. JMS dışı uygulamalar için dönüştürme yeteneği her zaman `GetMessage`Seçeneği, `GMO_CONVERT`Tayarı tarafından kullanılabilir olmuştur. Kuyruk yöneticisi dönüştürme yeteneği, V7.0tarihine kadar bir ileti alan bir JMS uygulaması için kullanılabilir değil.

Bir ileti gönderen bir JMS istemci uygulaması ile V7.0öncesinde kuyruk yöneticisi dönüştürmesini kullanabilirsiniz. JMS istemcisi biçimlendirilmiş bir kayıt oluşturur, iletide yer alan verilere karşılık gelen `CodedCharacterSetId`, `Encoding`ve `Format` özniteliklerini ayarlar. JMS dışı giriş uygulaması, `GMO_CONVERT`komutunu kullanarak iletiyi okur ve kullanıcı tarafından yazılan veri dönüştürme çıkışlarının çağrılmasına neden olur. Veri dönüştürme çıkışı, adı `Format` alanında ayarlanmış olan paylaşılan bir kitaptır.

V7.0' den bu yana, kuyruk yöneticisi JMS istemcilerine gönderilen iletileri dönüştürebilmekte. 7.0.0.0 ile 7.0.1.4 arasında, kuyruk yöneticisi dönüştürmesi her zaman JMS istemcileri için çağrılır. From 7.0.1.5, or from 7.0.1.4 with APAR IC72897 applied, queue manager conversion is controlled by setting the destination property, `WMQ_RECEIVE_CONVERSION`, to `WMQ_RECEIVE_CONVERSION_QMGR`, or `WMQ_RECEIVE_CONVERSION_CLIENT_MSG`. `WMQ_RECEIVE_CONVERSION_CLIENT_MSG` is the default setting, matching the behavior of WebSphere MQ V6.0, which did not support queue manager data conversion for JMS clients. Uygulama hedef ayarını değiştirebilir:

```
((MQDestination)destination).setIntProperty(
              WMQConstants.WMQ_RECEIVE_CONVERSION,
              WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
```

Ya da,

```
((MQDestination)destination).setReceiveConversion
             (WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
```

Şekil 152. Kuyruk yöneticisi veri dönüştürmesini etkinleştir

İstemci bir `consumer.receive` yöntemini çağırdığında, JMS istemcisi için kuyruk yöneticisi veri dönüştürme işlemi gerçekleşir. Metin verileri varsayılan olarak UTF-8 (1208) olarak dönüştürülür. Sonraki okuma ve alma yöntemleri, UTF-8' tan alınan verilerdeki metnin kodunu çözer ve iç Unicode kodlamalarında Java metin primitileri yaratır. UTF-8 is not the only target character set from queue manager data conversion. `WMQ_RECEIVE_CCSID` hedef özelliğini ayarlayarak farklı bir CCSID seçebilirsiniz.

Bir uygulama hedef ayarını da değiştirebilir, örneğin, bunu 437, DOS-US olarak ayarlamak gibi.

```
((MQDestination)destination).setIntProperty
            (WMQConstants.WMQ_RECEIVE_CCSID, 437);
```

Ya da,

```
((MQDestination)destination).setReceiveCCSID(437);
```

Şekil 153. Kuyruk yöneticisi dönüştürmesi için hedef kodlanmış karakter kümesini ayarla

WMQ_RECEIVE_CCSID 'nin değişmesinin nedeni özeldir; seçilen CCSID, JVM' de yaratılan metin nesnelerinde hiçbir fark yaratmaz. Ancak bazı platformlarda bazı JVM 'ler, ileltteki metnin CCSID' lerinden Unicode 'a dönüştürme işlemi kaldıramayabilir. Bu seçenek, iletide istemciye teslim edilen herhangi bir metin için CCSID ' yi seçme olanağı sağlar. Bazı JMS istemci altyapılarında, UTF-8içinde teslim edilen ileti metniyle ilgili sorunlar var.

JMS kodu, Şekil 154 sayfa 814içindeki C kodundaki kalın metin metnine eşdeğerdir.

```
gmo.Options = MQGMO_WAIT         /* wait for new messages           */
            | MQGMO_NO_SYNCPOINT /* no transaction                  */
            | MQGMO_CONVERT;     /* convert if necessary            */

     while (CompCode != MQCC_FAILED) {
     buflen = sizeof(buffer) - 1; /* buffer size available for GET   */
     memcpy(md.MsgId, MQMI_NONE, sizeof(md.MsgId));
     memcpy(md.CorrelId, MQCI_NONE, sizeof(md.CorrelId));
     md.Encoding       = MQENC_NATIVE;
     md.CodedCharSetId = MQCCSI_Q_MGR;

           MQGET(Hcon,          /* connection handle               */
           Hobj,                /* object handle                   */
           &md,                 /* message descriptor              */
           &gmo,                /* get message options             */
           buflen,              /* buffer length                   */
           buffer,              /* message buffer                  */
           &messlen,            /* message length                  */
           &CompCode,           /* completion code                 */
           &Reason);            /* reason code                     */
```

Şekil 154. Kod parçacığı: amqsget0.c

Not:

Kuyruk yöneticisi dönüştürmesi yalnızca, bilinen bir WebSphere MQ biçimine sahip ileti verileri üzerinde gerçekleştirilir. MQSTR, ya da MQCIH , önceden tanımlanmış bilinen biçimlere örneklerdir. Veri dönüştürme çıkışı sağladığınız sürece, bilinen bir biçim de kullanıcı tanımlı biçim de olabilir.

JMSTextMessage, JMSMapMessage ve JMSStreamMessageolarak oluşturulan iletiler bir MQSTR biçimine sahiptir ve kuyruk yöneticisi tarafından dönüştürülebilirler.

İlgili kavramlar

JMS ileti dönüştürme yaklaşımları
Bir dizi veri dönüştürme yaklaşımları JMS uygulama tasarımcılarına açıktır. Bu yaklaşımlar münhasır değildir; bazı uygulamalar bu yaklaşımların bir birleşimini kullanma olasılığının yüksek olduğu bir tür. Uygulamanız yalnızca metin alışverişi yapmışsa ya da yalnızca diğer JMS uygulamalarıyla ileti alışverişi yapmışsa, normalde veri dönüştürmeyi dikkate almayın. Veri dönüştürme işlemi sizin için otomatik olarak WebSphere MQtarafından gerçekleştirilir.

JMS istemcisi ileti dönüştürme ve kodlama
JMS istemcisi ileti dönüştürme ve kodlamasını yapmak için kullandığınız yöntemler, her dönüştürme tipine ilişkin kod örnekleriyle listelenir.

“Veri dönüştürme çıkışı çağrılıyor” sayfa 398

Veri dönüştürme çıkışı, bir MQGET çağrısının işlenmesi sırasında denetimi alan, kullanıcı tarafından yazılmış bir çıkışıdır.

İlgili görevler

JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et
Bir veri dönüştürme çıkışı tasarlamak ve oluşturmak için bu görevde önerilen adımları izleyin ve `JMSBytesMessage`eile JMS dışı bir uygulamayla iletileri değiş tokuş eden bir JMS istemcisi uygulaması. JMS dışı bir uygulamayla biçimlendirilmiş bir iletinin değiş tokası, veri dönüştürme çıkışı çağrılmadan ya da çağrılmadan gerçekleşebilir.

İlgili başvurular

JMS ileti tipleri ve dönüştürme
İleti tipi seçimi, ileti dönüştürmeye yaklaşımınızı etkiler. İleti dönüştürme ve ileti tipi etkileşimi, JMS ileti tipleri `JMSObjectMessage`, `JMSTextMessage`, `JMSMapMessage`, `JMSStreamMessage`eve `JMSBytesMessage`JMS ileti tipleri için açıklanmıştır.

JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et

Bir veri dönüştürme çıkışı tasarlamak ve oluşturmak için bu görevde önerilen adımları izleyin ve `JMSBytesMessage`eile JMS dışı bir uygulamayla iletileri değiş tokuş eden bir JMS istemcisi uygulaması. JMS dışı bir uygulamayla biçimlendirilmiş bir iletinin değiş tokası, veri dönüştürme çıkışı çağrılmadan ya da çağrılmadan gerçekleşebilir.

Başlamadan önce

You might be able to design a simpler solution to exchanging messages with a non-JMS application using a `JMSTextMessage`. Bu görevdeki adımları takip etmeden önce bu olasılığı ortadan kaldırın.

Bu görev hakkında

Bir JMS istemcisi, diğer JMS istemcileriyle değiş tokuş edilen JMS iletilerinin biçimlendirilmesi ayrıntılarına dahil edilmezse, daha kolay yazılır. İleti tipi `JMSTextMessage`, `JMSMapMessage`, `JMSStreamMessage`eya da `JMSObjectMessage`olduğu sürece, WebSphere MQ iletinin biçimlendirilmesi ayrıntılarının sonrasına bakar. WebSphere MQ , farklı platformlardaki kod sayfalarındaki ve sayısal kodlamadaki farklılıklarla ilgilidir.

JMS dışı uygulamalarla ileti alışverişi yapmak için bu ileti tiplerini kullanabilirsiniz. Bunu yapmak için, bu iletilerin JMS için WebSphere MQ sınıfları tarafından nasıl oluşturulacağını anlamanız gerekir. JMS dışı uygulamayı iletileri yorumlamak için değiştirebilir; bkz. “JMS iletilerinin WebSphere MQ iletilerine eşlenmesi” sayfa 778.

Bu ileti tiplerinden birini kullanmanın yararı, JMS istemci programlamasıdır. Bu programlama, ileti alışverişi yaptığı uygulama tipine bağlıdır. Bir dezavantaj, başka bir program için bir değişiklik gerektirmesi ve diğer programı değiştiremeyebilirsiniz.

Alternatif bir yaklaşım, var olan ileti biçimleriyle başa çıkabilen bir JMS istemcisi uygulaması yazmadır. Sık sık varolan iletiler değişmez biçimdir ve biçimlenmemiş veri, metin ve sayılardan oluşan bir karışım içerir. JMS dışı uygulamalarla biçimlendirilmiş kayıtlar değiş tokuş edebilen bir JMS istemcisi oluşturmak için başlangıç noktası olarak bu görevdeki adımları ve “Writing classes to encapsulate a record layout in a JMSBytesMessage” sayfa 818içindeki örnek JMS istemcisini kullanın.

Yordam

1. Kayıt yerleşim düzenini tanımlayın ya da önceden tanımlı WebSphere MQ üstbilgi sınıflarından birini kullanın.

   Önceden tanımlı WebSphere MQ üstbilgilerini işlemek için, WebSphere MQ ileti üstbilgilerini işlemebaşlıklı konuya bakın.

   Şekil 155 sayfa 816 , veri dönüştürme yardımcı programı tarafından işlenebilen, kullanıcı tanımlı, sabit uzunluklu kayıt düzenine bir örnektir.
2. Veri dönüştürme çıkışısını yaratın.

Veri dönüştürme çıkışı yazmak için Writing a data-conversion exit program (Veri dönüştürme çıkış programı yazma) yönergelerini izleyin.

To try out the example in “Writing classes to encapsulate a record layout in a JMSBytesMessage” sayfa 818, name the data conversion exit MYRECORD.

3. Kayıt yerleşim düzenini kapsüllemek ve kayıt göndermek ve almak için Java sınıflarını yazın. Alabileceğiniz iki yaklaşım şunlardır:
   - Bu okumaya bir sınıf yazın ve kaydı içeren `JMSBytesMessage` ' ı yazar; bkz. “Writing classes to encapsulate a record layout in a JMSBytesMessage” sayfa 818.
   - Kaydın veri yapısını tanımlamak için `com.ibm.mq.header.Header` sınıfını genişleten bir sınıf yazın; bkz. Yeni üstbilgi tipleri için sınıflar oluşturma.
4. İletileri değiş tokuş etmek için hangi kodlanmış karakter takısının belirleneceğine karar verin.

   Bkz. “İleti dönüştürmesi için bir yaklaşım seçilmesi: alıcı iyi olur” sayfa 797.
5. Configure the destination to exchange MQ-type messages, without a JMS MQRFH2 header.

   Hem gönderme, hem de alma hedefi, MQtipi iletileri değiş tokuş edecek şekilde yapılandırılmalıdır. Aynı hedefi her iki gönderme ve alma için de kullanabilirsiniz.

   Uygulama, hedef ileti gövdesi özelliğini geçersiz kılabilir:

   ```
   ((MQDestination)destination).setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_MQ);
   ```

   “Writing classes to encapsulate a record layout in a JMSBytesMessage” sayfa 818 içindeki örnek, hedef ileti gövdesi özelliğini geçersiz kılar ve bir MQstili iletinin gönderilmesine olanak sağlar.
6. JMS ve JMS dışı uygulamalar ile çözümü test edin

   Veri dönüştürme çıkışını sınamak için kullanılabilecek yararlı araçlar şunlardır:
   - `amqsgetc0.c` örnek programı, JMS istemcisi tarafından gönderilen bir iletinin alınmasını test etmek için kullanışlıdır. Şekil 156 sayfa 817içinde, örnek üstbilgisini (`RECORD.h`) kullanmak için önerilen değişiklikleri görün. With the modifications, `amqsgetc0.c` receives a message sent by the example JMS client, `TryMyRecord.java`; see “Writing classes to encapsulate a record layout in a JMSBytesMessage” sayfa 818.
   - Örnek WebSphere MQ göz atma programı, `amqsbcg0.c`, ileti üstbilgisinin içeriğini, JMS üstbilgisini (MQRFH2) ve ileti içeriğini incelemek için kullanışlıdır.
   - The **`rfhutil`** program, previously available in SupportPac IH03, allows test messages to be captured and stored in files, and then used to drive Message Flows. Çıkış iletileri çeşitli biçimlerde de okunabilir ve görüntülenebilir. Bu biçimler, iki XML tipini ve bir COBOL copybook ile eşleşmeyi içerir. Veriler EBCDIC ya da ASCII ' de olabilir. İleti gönderilmeden önce iletiye bir RFH2 üstbilgisi eklenebilir.

   Değiştirilen `amqsgetc0.c` örnek programını kullanarak ileti almaya çalışırsanız ve neden kodu `2080`ile bir hata alırsanız, iletinin MQRFH2olup olmadığını denetleyin. Bu değişiklikler, iletinin MQRFH2no ' u belirten bir hedefe gönderildiğini varsayar.

Örnekler

```
struct RECORD { MQCHAR StrucID[4];
                MQLONG Version;
                MQLONG StructLength;
                MQLONG Encoding;
                MQLONG CodeCharSetId;
                MQCHAR Format[8];
                MQLONG Flags;
                MQCHAR RecordData[32];
    };
```

Şekil 155. RECORD.h

- RECORD.h veri yapısını bildirme

```
struct tagRECORD {
   MQCHAR4    StrucId;
   MQLONG     Version;
   MQLONG     StrucLength;
   MQLONG     Encoding;
   MQLONG     CCSID;
   MQCHAR8    Format;
   MQLONG     Flags;
   MQCHAR32   RecordData;
 };
typedef struct tagRECORD RECORD;
   typedef RECORD MQPOINTER PRECORD;
   RECORD record;
   PRECORD pRecord = &(record);
```

- Modify the MQGET call to use RECORD,

  1. Değişiklikten önce:

```
    MQGET(Hcon,            /* connection handle     */
          Hobj,            /* object handle         */
          &md,             /* message descriptor    */
          &gmo,            /* get message options   */
          buflen,          /* buffer length         */
          buffer,          /* message buffer        */
          &messlen,        /* message length        */
          &CompCode,       /* completion code       */
          &Reason);        /* reason code           */
```

  2. Değişiklikten sonra:

```
    MQGET(Hcon,            /* connection handle     */
          Hobj,            /* object handle         */
          &md,             /* message descriptor    */
          &gmo,            /* get message options   */
          sizeof(RECORD),  /* buffer length         */
          pRecord,         /* message buffer        */
          &messlen,        /* message length        */
          &CompCode,       /* completion code       */
          &Reason);        /* reason code           */
```

- Yazdırma deyimini değiştirme,

  1. Başlangıç:

```
buffer[messlen] = '\0';            /* add terminator          */
printf("message <%s>\n", buffer);
```

  2. Hedef:

```
/* buffer[messlen] = '\0';            add terminator          */
printf("ccsid <%d>, flags <%d>, message <%32.32s>\n \0",
        md.CodedCharSetId, record.Flags, record.RecordData);
```

Şekil 156. Modify amqsget0.c

İlgili kavramlar

JMS ileti dönüştürme yaklaşımları

Bir dizi veri dönüştürme yaklaşımları JMS uygulama tasarımcılarına açıktır. Bu yaklaşımlar münhasır değildir; bazı uygulamalar bu yaklaşımların bir birleşimini kullanma olasılığının yüksek olduğu bir tür. Uygulamanız yalnızca metin alışverişi yapmışsa ya da yalnızca diğer JMS uygulamalarıyla ileti alışverişi yapmışsa, normalde veri dönüştürmeyi dikkate almayın. Veri dönüştürme işlemi sizin için otomatik olarak WebSphere MQtarafından gerçekleştirilir.

JMS istemcisi ileti dönüştürme ve kodlama

JMS istemcisi ileti dönüştürme ve kodlamasını yapmak için kullandığınız yöntemler, her dönüştürme tipine ilişkin kod örnekleriyle listelenir.

Kuyruk yöneticisi verileri dönüştürme

Kuyruk yöneticisi verileri dönüştürme her zaman JMS istemcilerinden ileti alan JMS dışı uygulamalar için kullanılabilir. V7.0' dan bu yana, ileti alan JMS istemcileri de kuyruk yöneticisi veri dönüştürmesini kullanır. 7.0.1.5ya da APAR IC72897ile 7.0.1.4 , kuyruk yöneticisi veri dönüştürme isteğe bağlıdır.

İlgili başvurular

JMS ileti tipleri ve dönüştürme

İleti tipi seçimi, ileti dönüştürmeye yaklaşımınızı etkiler. İleti dönüştürme ve ileti tipi etkileşimi, JMS ileti tipleri `JMSObjectMessage`, `JMSTextMessage`, `JMSMapMessage`, `JMSStreamMessage`eve `JMSBytesMessage`JMS ileti tipleri için açıklanmıştır.

Dönüştürme-çıkış kodu yaratmak için kullanılan yardımcı program

Writing classes to encapsulate a record layout in a `JMSBytesMessage`

Bu görevin amacı, bir `JMSBytesMessage`eiçindeki veri dönüştürmenin ve sabit kayıt yerleşim düzeninin nasıl birleştirileceğini, örneğin, nasıl birleştirileceğini keşfetmek için kullanılır. Bu görevde, bir `JMSBytesMessage`eiçinde örnek bir kayıt yapısı değiştirmek için bazı Java sınıfları oluştursunuz. Diğer kayıt yapılarını değiş tokuş etmek için sınıf yazmak için örneği değiştirebilirsiniz.

A `JMSBytesMessage` is the best choice of JMS message type to exchange mixed data type records with non-JMS programs. JMS sağlayıcısıyla ileti gövdesine eklenen hiçbir ek veri yok. Bu nedenle, bir JMS istemcisi programı var olan bir IBM WebSphere MQ programıyla çalışabiliyorsa, kullanılacak ileti tipinin en iyi seçimi olur. Bir `JMSBytesMessage` kullanılmasındaki ana sınama, diğer program tarafından beklenen kodlama ve karakter kümesiyle eşleşen bir değer sağlar. Bir çözüm, kaydı sarmalayan bir sınıf oluşturmaktan başka bir şey değildir. A class that encapsulates reading and writing a `JMSBytesMessage`, for a specific record type, makes it easier to send and receive fixed-format records in a JMS program. Arabirimin soyut bir sınıftaki soysal yönlerini yakalayarak, çözümün büyük bölümü farklı kayıt biçimleri için yeniden kullanılabilir. Soyut soysal sınıfı genişleten sınıflarda farklı kayıt biçimleri uygulanabilir.

Alternatif bir yaklaşım, `com.ibm.mq.headers.Header` sınıfını genişletebilmektedir. `Header` sınıfının, daha açıklayıcı bir şekilde kayıt biçimi oluşturmak için `addMQLONG`gibi yöntemleri vardır. `Header` sınıfının kullanılmasının bir dezavantajı ise, öznitelikler daha karmaşık bir yorumlayıcı arabirimi kullanır ve ayarlanıyor. Her iki yaklaşım da aynı sayıda uygulama kodunda sonuç elde eder.

Bir `JMSBytesMessage` , her bir kayıt aynı biçimi, kodlanmış karakter kümesini ve kodlamayı kullanmadığı sürece, bir iletide `MQRFH2`' un yanı sıra yalnızca tek bir biçimi sarsabilir. Bir `JMSBytesMessage` 'in biçim, kodlama ve karakter kümesi, `MQRFH2`' in ardından gelen tüm iletilerin özellikleridir. Örnek, `JMSBytesMessage` ' un yalnızca bir kullanıcı kaydı içerdiği varsayımına yazılır.

Başlamadan önce

1. Beceri düzeyiniz: Java programlama ve JMS ile ilgili bilgi sahibi olmanız gerekir. Java geliştirme ortamını ayarlama hakkında herhangi bir yönerge sağlanmaz. It is advantageous to have written a program to exchange a `JMSTextMessage`, `JMSStreamMessage`, or `JMSMapMessage`. Daha sonra, `JMSBytesMessage`kullanarak bir ileti alışverişi arasındaki farkları görebilirsiniz.
2. Örnek, IBM WebSphere MQ V7.0' ı gerektirir.
3. Örnek, Eclipse Workbench 'in Java perspektifi kullanılarak yaratıldı. JRE 6.0 ya da üstünü gerektirir. Java sınıflarını geliştirmek ve çalıştırmak için IBM WebSphere MQ Gezgini 'nde Java perspektifini kullanabilirsiniz. Diğer bir seçenek olarak, kendi Java geliştirme ortamınızı kullanabilirsiniz.
4. IBM WebSphere MQ Explorer 'ın kullanılması, test ortamını ayarlamayı ve hata ayıklamayı, komut satırı yardımcı programlarını kullanmaktan daha basitleştirir.

Bu görev hakkında

İki sınıf oluşturma yoluyla yönlendirildiniz: `RECORD` ve `MyRecord`. Bu iki sınıf birlikte, sabit biçimli bir kaydı sarsalıyor. Öznitelikleri elde etmek ve ayarlamak için yöntemleri var. Get (alma) yöntemi, kaydı `JMSBytesMessage` 'den okur ve put yöntemi bir kaydı `JMSBytesMessage`' e yazar.

Görevin amacı, yeniden kullanabileceğiniz bir üretim kalitesi sınıfı yaratmamaktadır. Kendi sınıflarınıza başlamak için görevdeki örnekleri kullanmayı tercih edebilirsiniz. The purpose of the task is to provide

you with guidance notes, primarily about using character sets, formats, and encoding, when using a JMSBytesMessage. Sınıfları oluşturmadaki her adım, bazen gözden kaçırılan JMSBytesMessage' u kullanma yönlerinin açıklanandır ve açıklarıdır.

RECORD sınıfı soyut (abstract) ve bir kullanıcı kaydı için bazı ortak alanları tanımlar. Ortak alanlar, göz yakalayıcısına, sürüme ve uzunluk alanına sahip olmanın standart IBM WebSphere MQ üstbilgi düzenine modellenir. Birçok IBM WebSphere MQ üstbilgisinde bulunan kodlama, karakter kümesi ve biçim alanları atılır. Başka bir üstbilgi kullanıcı tanımlı biçimi izleyemez. RECORD sınıfını genişleten MyRecord sınıfı, kaydı ek kullanıcı alanlarıyla birlikte genişleterek gerçekleştirir. Sınıflar tarafından yaratılan bir JMSBytesMessagedeğeri, kuyruk yöneticisi veri dönüştürme çıkışı tarafından işlenebilir.

"Örneği çalıştırmak için kullanılan sınıflar" sayfa 824 includes a full listing of RECORD and MyRecord. Ayrıca, RECORD ve MyRecord' yi sınamak için ek "iskeleler" sınıflarının listelemeleri de yer alır. Ek sınıflar şunlardır:

TryMyRecord
RECORD ve MyRecord' yi test etmek için kullanılan ana program.

EndPoint
Tek bir sınıftaki JMS bağlantısını, hedefini ve oturumunu kapatan soyut bir sınıf. Arabirimi yalnızca RECORD ve MyRecord sınıflarının test edilmesi gereksinimlerini karşılar. JMS uygulamaları yazmak için oluşturulmuş bir tasarım deseni değildir.

Not: Endpoint sınıfı, bir hedef oluşturduktan sonra bu kod satırını içerir:

```
((MQDestination)destination).setReceiveConversion
                            (WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
```

V7.0'da V7.0.1.5' de kuyruk yöneticisi dönüşümünü açmak gerekir. Varsayılan değer olarak devre dışı bırakılır. V7.0' da en çok V7.0.1.4 kuyruk yöneticisi dönüşümü varsayılan olarak etkindir ve bu kod satırı bir hataya neden olur.

MyProducer ve MyConsumer
Classes that extend EndPoint, and create a MessageConsumer and MessageProducer, connected and ready to accept requests.

Tüm sınıflar bir JMSBytesMessage' de veri dönüştürmeyi nasıl kullanacağını anlamak için, tüm sınıfları oluşturabileceğiniz ve deneyebileceğiniz eksiksiz bir uygulama oluşturur.

Yordam

1. Bir IBM WebSphere MQ üstbilgisindeki standart alanları bir varsayılan oluşturucuyla sarmalamak için bir soyut sınıf yaratın. Daha sonra, üstbilgiyi gereksinimlerinize göre uyarlamak için sınıfı genişletiyorsunuz.

```
public abstract class RECORD implements Serializable {
    private static final long serialVersionUID = -1616617232750561712L;
    protected final static int UTF8 = 1208;
    protected final static int MQLONG_LENGTH = 4;
    protected final static int RECORD_STRUCT_ID_LENGTH = 4;
    protected final static int RECORD_VERSION_1 = 1;
    protected final String RECORD_STRUCT_ID = "BLNK";
    protected final String RECORD_TYPE = "BLANK   ";
    private String structID = RECORD_STRUCT_ID;
    private int version = RECORD_VERSION_1;
    private int structLength = RECORD_STRUCT_ID_LENGTH + MQLONG_LENGTH * 2;
    private int headerEncoding = WMQConstants.WMQ_ENCODING_NATIVE;
    private String headerCharset = "UTF-8";
    private String headerFormat = RECORD_TYPE;

    public RECORD() {
        super();
    }
```

Not:

a. The attributes, structID to nextFormat, are listed in the order they are laid out in a standard IBM WebSphere MQ message header.

b. The attributes, `format`, `messageEncoding`, and `messageCharset`, describe the header itself, and are not part of the header.

c. Kaydın kodlanmış karakter takımı tanıtıcısını ya da karakter kümesini saklanıp saklanmayacağınıza karar vermeniz gerekir. Java karakter kümeleri ve IBM WebSphere MQ iletileri, kodlanmış karakter takımı tanıtıcılarını kullanır. Örnek kod karakter kümelerini kullanır.

d. `int` is serialized to `MQLONG` by IBM WebSphere MQ. `MQLONG` , 4 bayttır.

2. Özel öznitelikler için alıcıları ve ayarlayıcıları yaratın.

a) Alıcıları yarat ya da oluştur:

```
    public String getHeaderFormat() { return headerFormat; }
    public int getHeaderEncoding() { return headerEncoding;    }
    public String getMessageCharset() { return headerCharset; }
    public int getMessageEncoding() { return headerEncoding; }
    public String getStructID() { return structID; }
    public int getStructLength() { return structLength; }
    public int getVersion() { return version; }
```

b) Ayarlayıcılar yarat ya da oluştur:

```
    public void setHeaderCharset(String charset) {
        this.headerCharset = charset; }
    public void setHeaderEncoding(int encoding) {
        this.headerEncoding = encoding; }
    public void setHeaderFormat(String headerFormat) {
        this.headerFormat = headerFormat; }
    public void setStructID(String structID) {
        this.structID = structID; }
    public void setStructLength(int structLength) {
        this.structLength = structLength; }
    public void setVersion(int version) {
        this.version = version; }
}
```

3. Create a constructor to create a RECORD instance from a `JMSBytesMessage`.

```
public RECORD(BytesMessage message) throws JMSException, IOException,
       MQDataException {
    super();
    setHeaderCharset(message.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET));
    setHeaderEncoding(message.getIntProperty(WMQConstants.JMS_IBM_ENCODING));
    byte[] structID = new byte[RECORD_STRUCT_ID_LENGTH];
    message.readBytes(structID, RECORD_STRUCT_ID_LENGTH);
    setStructID(new String(structID, getMessageCharset()));
    setVersion(message.readInt());
    setStructLength(message.readInt());
}
```

Not:

a. The `messageCharset` and `messageEncoding`, are captured from the message properties, as they override the values set for the destination. `format` güncellenmedi. Örnek, hata denetimi yapmaz. `Record(BytesMessage)` oluşturucusu çağrılırsa, `JMSBytesMessage` ' in RECORD tipinde bir ileti olduğu varsayılır. The line "`setStructID(new String(structID, getMessageCharset()))`" sets the eye catcher.

b. RECORD eşgörünümünde belirlenen varsayılan değerlerin güncellenmesi sırasında, bu yöntemi tamamlayan kodun satırlarında, satırdaki alanları diziselden geri çevirme işlemi gerçekleştirin.

4. Üstbilgi alanlarını bir `JMSBytesMessage`değerine yazmak için bir put yöntemi yaratın.

```
protected BytesMessage put(MyProducer myProducer) throws IOException,
        JMSException, UnsupportedEncodingException {
    setHeaderEncoding(myProducer.getEncoding());
    setHeaderCharset(myProducer.getCharset());
    myProducer.setMQClient(true);
    BytesMessage bytes = myProducer.session.createBytesMessage();
    bytes.setStringProperty(WMQConstants.JMS_IBM_FORMAT, getHeaderFormat());
    bytes.setIntProperty(WMQConstants.JMS_IBM_ENCODING, getHeaderEncoding());
    bytes.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET,
                         myProducer.getCCSID());
    bytes.writeBytes(String.format("%1$-" + RECORD_STRUCT_ID_LENGTH + "."
```

```
                        + RECORD_STRUCT_ID_LENGTH + "s", getStructID())
                        .getBytes(getMessageCharset()), 0, RECORD_STRUCT_ID_LENGTH);
        bytes.writeInt(getVersion());
        bytes.writeInt(getStructLength());
        return bytes;
    }
```

Not:

a. `MyProducer` , JMS `Connection`, `Destination`, `Session`ve `MessageProducer` JMS ' lerini tek bir sınıf içinde sarmadır. Daha sonra kullanılanJMSTüketiciMyConsumer, JMS `Connection`, `Destination`, `Session`ve `MessageConsumer` JMS ' lerini tek bir sınıfta sarmalıyor.

b. Bir `JMSBytesMessage`için, kodlama `Native`' den başka bir kodsa, kodlamada kodlamanın ayarlanması gerekir. The destination encoding is copied to the message encoding attribute, `JMS_IBM_CHARACTER_SET`, and saved as an attribute of the `RECORD` class.

   i) "`setMessageEncoding(myProducer.getEncoding());`" , hedef kodlamayı almak için "`(((MQDestination) destination).getIntProperty(WMQConstants.WMQ_ENCODING));`" ' i çağırır.

   ii) "`Bytes.setIntProperty(WMQConstants.JMS_IBM_ENCODING, getMessageEncoding());`" , ileti kodlamasını ayarlar.

c. Metni bayt olarak dönüştürmek için kullanılan karakter takımı hedeften alınır ve `RECORD` sınıfının bir özniteliği olarak kaydedilir. Bir `JMSBytesMessage`yazılırken JMS için IBM WebSphere MQ sınıfları tarafından kullanılmadığı için, iletide bu ad belirlenmez.

   "`messageCharset = myProducer.getCharset();`" çağrıları

```
    public String getCharset() throws UnsupportedEncodingException,
            JMSException {
        return CCSID.getCodepage(getCCSID());
    }
```

   Kodlanmış karakter takımı tanıtıcısından ayarlanan Java karakterini alır.

   "`CCSID.getCodepage(ccsid)`" is in the package `com.ibm.mq.headers`. The `ccsid` is obtained from another method in `MyProducer`, which queries the destination:

```
    public int getCCSID() throws JMSException {
        return (((MQDestination) destination)
                .getIntProperty(WMQConstants.WMQ_CCSID));
    }
```

d. "`myProducer.setMQClient(true);`" , istemci tipine ilişkin hedef ayarı geçersiz kılar ve bunu bir IBM WebSphere MQ MQI istemcisine zorlar. Bir yönetim yapılandırma hatasını gizlediğinden, bu kod satırını atlamayı tercih edebilirsiniz.

   "`myProducer.setMQClient(true);`" çağrıları:

```
((MQDestination) destination).setTargetClient(WMQConstants.WMQ_TARGET_DEST_MQ); }
if (!getMQDest()) setMQBody();
```

   The code has the side-effect of setting the IBM WebSphere MQ body style to unspecified, if it must override a setting of JMS.

   Not:

   The IBM WebSphere MQ classes for JMS write the format, encoding, and character set identifier of the message into the message descriptor, `MQMD`, or into the JMS header, `MQRFH2`. Bu, iletinin bir IBM WebSphere MQ biçemi gövdesi olup olmadığına bağlıdır. `MQMD` alanlarını el ile ayarlamayın.

   İleti tanımlayıcı özelliklerini el ile ayarlamak için bir yöntem vardır. `JMS_IBM_MQMD_*` özelliklerini kullanır. You must set the destination property, `WMQ_MQMD_WRITE_ENABLED` to set the `JMS_IBM_MQMD_*` properties:

```
((MQDestination)destination).setMQMDWriteEnabled(true);
```

Özellikleri okumak için hedef özelliğini (WMQ_MQMD_READ_ENABLED) ayarlamanız gerekir.

JMS_IBM_MQMD_* ' ı yalnızca tüm ileti bilgi yükü üzerinde tam denetim ele aldıysanız kullanın. JMS_IBM_* özelliklerinden farklı olarak, JMS_IBM_MQMD_* özellikleri JMS için IBM WebSphere MQ sınıflarının JMS iletisini nasıl oluşturacağını denetmez. JMS iletisinin özellikleriyle çakışan ileti tanımlayıcı özellikleri yaratmak mümkündür.

e. Yöntemi, sınıftaki öznitelikleri, iletteki alanlar olarak serileştiren kod satırlarını içerir.

Dizgi öznitelikleri boşluklarla dolduruldu. Dizgiler, kayıt için tanımlanan karakter takımı kullanılarak bayt 'a dönüştürülür ve ileti alanlarının uzunluğuna kısaltılır.

5. İçe aktarma deyimlerini ekleyerek sınıfı tamamlayın.

```
package com.ibm.mq.id;
import java.io.IOException;
import java.io.Serializable;
import java.io.UnsupportedEncodingException;
import javax.jms.BytesMessage;
import javax.jms.JMSException;
import com.ibm.mq.constants.MQConstants;
import com.ibm.mq.headers.MQDataException;
import com.ibm.msg.client.wmq.WMQConstants;
```

6. Ek alanları dahil etmek için RECORD sınıfını genişletmek için bir sınıf oluşturun. Varsayılan bir oluşturucu ekleyin.

```
public class MyRecord extends RECORD {
    private static final long serialVersionUID = -370551723162299429L;
    private final static int FLAGS = 1;
    private final static String STRUCT_ID = "MYRD";
    private final static int DATA_LENGTH = 32;
    private final static String FORMAT = "MYRECORD";
    private int flags = FLAGS;
    private String recordData = "ABCDEFGHIJKLMNOPQRSTUVWXYZ012345";

    public MyRecord() {
        super();
        super.setStructID(STRUCT_ID);
        super.setHeaderFormat(FORMAT);
        super.setStructLength(super.getStructLength() + MQLONG_LENGTH
                + DATA_LENGTH);
    }
```

Not:

a. RECORD alt sınıfı, MyRecord, gözün yakalayıcıya, biçimini ve uzunluğunun özelleştirilmesini sağlar.

7. Alıcıları ve ayarlayıcıları yaratın ya da oluşturun.

a) Alıcıları yarat:

```
    public int getFlags() { return flags; }
    public String getRecordData() { return recordData; }Â .
```

b) Ayarlayıcıları yaratın:

```
    public void setFlags(int flags) {
        this.flags = flags; }
    public void setRecordData(String recordData) {
        this.recordData = recordData; }
}
```

8. Create a constructor to create a MyRecord instance from a JMSBytesMessage.

```
    public MyRecord(BytesMessage message) throws JMSException, IOException,
            MQDataException {
        super(message);
        setFlags(message.readInt());
        byte[] recordData = new byte[DATA_LENGTH];
        message.readBytes(recordData, DATA_LENGTH);
        setRecordData(new String(recordData, super.getMessageCharset()));
    }
```

Not:

a. Standart ileti şablonunu oluşturan alanlar, RECORD sınıfı tarafından önce okunur.

b. The `recordData` text is converted to `String` using the character set property of the message.

9. Bir tüketiciden ileti almak ve yeni bir `MyRecord` yönetim ortamı yaratmak için durağan bir yöntem yaratın.

```
public static MyRecord get(MyConsumer myConsumer) throws JMSException,
        MQDataException, IOException {
    BytesMessage message = (BytesMessage) myConsumer.receive();
    return new MyRecord(message);
}
```

Not:

a. In the example, for brevity, the `MyRecord(BytesMessage)` constructor is called from the static get method. Genellikle, yeni bir `MyRecord` yönetim ortamı yaratmaktan ileti almayı birbirinden ayırabilirsiniz.

10. Müşteri alanlarını, ileti üstbilgisi içeren bir `JMSBytesMessage` ' a eklemek için bir put yöntemi oluşturun.

```
public BytesMessage put(MyProducer myProducer) throws JMSException,
        IOException {
    BytesMessage bytes = super.put(myProducer);
    bytes.writeInt(getFlags());
    bytes.writeBytes(String.format("%1$-" + DATA_LENGTH + "."
                    + DATA_LENGTH + "s",getRecordData())
                    .getBytes(super.getMessageCharset()), 0, DATA_LENGTH);
    myProducer.send(bytes);
    return bytes;
}
```

Not:

a. Koddaki yöntem çağrıları, MyRecord sınıfındaki öznitelikleri, iletteki alanlar olarak diziselleştirir.

- `recordData` `String` özniteliği boşluklarla doldurulur, kayıt için tanımlanan karakter kümesi kullanılarak baytlara dönüştürülür ve `RecordData` alanlarının uzunluğuna kısaltılır.

11. İçerme deyimlerini ekleyerek sınıfı tamamlayın.

```
package com.ibm.mq.id;
import java.io.IOException;
import javax.jms.BytesMessage;
import javax.jms.JMSException;
import com.ibm.mq.headers.MQDataException;
```

Sonuçlar

Sonuçlar:

- `TryMyRecord` sınıfını çalıştıktan sonra elde edilen sonuçlar:
  - İletinin kodlanmış karakter takımı 37 'de gönderilmesi ve kuyruk yöneticisi dönüştürme çıkışısının kullanılması:

```
Out flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 546 CCSID 37 MQ true
Out flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 546 CCSID 37 MQ true
In  flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 273 CCSID UTF-8
```

  - İleti, kodlanmış karakter takımı 37 'de gönderiliyor ve kuyruk yöneticisi dönüştürme çıkışı kullanılarak *yok* :

```
Out flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 546 CCSID 37 MQ true
Out flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 546 CCSID 37 MQ true
In  flags 1 text ABCDEFGHIJKLMNOPQRSTUVWXYZ012345 Encoding 546 CCSID IBM037
```

- The results from modifying the `TryMyRecord` class not to receive the message, and instead receiving it using the modified `amqsget0.c` sample. The modified sample accepts a formatted record; see Şekil 156 sayfa 817 in “JMS dışı bir uygulamayla biçimlendirilmiş bir kayıt değiş tokuş et” sayfa 815.
  - İletinin kodlanmış karakter takımı 37 'de gönderilmesi ve kuyruk yöneticisi dönüştürme çıkışısının kullanılması:

    ```
    Sample AMQSGET0 start
    ccsid <850>, flags <1>, message <ABCDEFGHIJKLMNOPQRSTUVWXYZ012345>
    no more messages
    Sample AMQSGET0 end
    ```

  - İleti, kodlanmış karakter takımı 37 'de gönderiliyor ve kuyruk yöneticisi dönüştürme çıkışı kullanılarak *yok* :

    ```
    Sample AMQSGET0 start
    MQGET ended with reason code 2110
    ccsid <37>, flags <1>, message <--+-+ãÃ++ÐÊËÈiÐÎÐ+ÔÒõõµþÞÚ-±=¾¶§>
    no more messages
    Sample AMQSGET0 end
    ```

Örnek ve deneyi farklı kod sayfaları ve bir veri dönüştürme çıkışı ile denemek için. Java sınıflarını oluşturun, IBM WebSphere MQ' u yapılandırın ve ana programı çalıştırın; `TryMyRecord`; bkz. Şekil 157 sayfa 825.

1. Örneği çalıştırmak için IBM WebSphere MQ ve JMS ' yi yapılandırın. The instructions are for running the example on Pencereler.

   1. Kuyruk yöneticisi yaratılması

   ```
   crtmqm -sa -u SYSTEM.DEAD.LETTER.QUEUE QM1
   strmqm QM1
   ```

   2. Bir kuyruk yaratın

   ```
   echo DEFINE QL('Q1') REPLACE | runmqsc QM1
   ```

   3. Bir JNDI dizini yaratın

   ```
   cd c:\
   md JNDI-Directory
   ```

   4. JMS bin dizinine geç

   JMS Denetimi programı buradan çalıştırılmalıdır. Yol şöyledir: *MQ_INSTALLATION_PATH*`\java\bin`.

   5. `JMSQM1Q1.txt`adlı bir dosyada aşağıdaki JMS tanımlarını yaratın.

   ```
   DEF CF(QM1) PROVIDERVERSION(7) QMANAGER(QM1)
   DEF Q(Q1) CCSID(37) ENCODING(RRR) MSGBODY(MQ) QMANAGER(QM1) QUEUE(Q1) TARGCLIENT(MQ)
   VERSION(7)
   END
   ```

   6. JMS kaynaklarını yaratmak için JMSAdmin programını çalıştırın.

   ```
   JMSAdmin < JMSQM1Q1.txt
   ```

2. IBM WebSphere MQ Gezgini 'ni kullanarak yarattığınız tanımlara göz atabilir, bunları değiştirebilir ve bunlara göz atabilirsiniz.
3. `TryMyRecord`komutunu çalıştırın.

Örneği çalıştırmak için kullanılan sınıflar

The classes listed in figures Şekil 157 sayfa 825 to Şekil 162 sayfa 829 are also available in a compressed file; download jm25529_.zip or jm25529_.tar.gz.

```
package com.ibm.mq.id;
public class TryMyRecord {
    public static void main(String[] args) throws Exception {
        MyProducer producer = new MyProducer();
        MyRecord outrec = new MyRecord();
        System.out.println("Out flags " + outrec.getFlags() + " text "
                + outrec.getRecordData() + " Encoding "
                + producer.getEncoding() + " CCSID " + producer.getCCSID()
                + " MQ " + producer.getMQDest());
        outrec.put(producer);
        System.out.println("Out flags " + outrec.getFlags() + " text "
                + outrec.getRecordData() + " Encoding "
                + producer.getEncoding() + " CCSID " + producer.getCCSID()
                + " MQ " + producer.getMQDest());
        MyRecord inrec = MyRecord.get(new MyConsumer());
        System.out.println("In  flags " + inrec.getFlags() + " text "
                + inrec.getRecordData() + " Encoding "
                + inrec.getMessageEncoding() + " CCSID "
                + inrec.getMessageCharset());
    }
}
```

Şekil 157. TryMyRecord

```
package com.ibm.mq.id;
import java.io.IOException;
import java.io.Serializable;
import java.io.UnsupportedEncodingException;
import javax.jms.BytesMessage;
import javax.jms.JMSException;
import com.ibm.mq.constants.MQConstants;
import com.ibm.mq.headers.MQDataException;
import com.ibm.msg.client.wmq.WMQConstants;

public abstract class RECORD implements Serializable {
    private static final long serialVersionUID = -1616617232750561712L;
    protected final static int UTF8 = 1208;
    protected final static int MQLONG_LENGTH = 4;
    protected final static int RECORD_STRUCT_ID_LENGTH = 4;
    protected final static int RECORD_VERSION_1 = 1;
    protected final String RECORD_STRUCT_ID = "BLNK";
    protected final String RECORD_TYPE = "BLANK   ";
    private String structID = RECORD_STRUCT_ID;
    private int version = RECORD_VERSION_1;
    private int structLength = RECORD_STRUCT_ID_LENGTH + MQLONG_LENGTH * 2;
    private int headerEncoding = WMQConstants.WMQ_ENCODING_NATIVE;
    private String headerCharset = "UTF-8";
    private String headerFormat = RECORD_TYPE;

    public RECORD() {
        super();
    }

    public RECORD(BytesMessage message) throws JMSException, IOException,
            MQDataException {
        super();
        setHeaderCharset(message.getStringProperty(WMQConstants.JMS_IBM_CHARACTER_SET));
        setHeaderEncoding(message.getIntProperty(WMQConstants.JMS_IBM_ENCODING));
        byte[] structID = new byte[RECORD_STRUCT_ID_LENGTH];
        message.readBytes(structID, RECORD_STRUCT_ID_LENGTH);
        setStructID(new String(structID, getMessageCharset()));
        setVersion(message.readInt());
        setStructLength(message.readInt());
    }

    public String getHeaderFormat() { return headerFormat; }
    public int getHeaderEncoding() { return headerEncoding;    }
    public String getMessageCharset() { return headerCharset; }
    public int getMessageEncoding() { return headerEncoding; }
    public String getStructID() { return structID; }
    public int getStructLength() { return structLength; }
    public int getVersion() { return version; }


    protected BytesMessage put(MyProducer myProducer) throws IOException,
            JMSException, UnsupportedEncodingException {
        setHeaderEncoding(myProducer.getEncoding());
        setHeaderCharset(myProducer.getCharset());
        myProducer.setMQClient(true);
        BytesMessage bytes = myProducer.session.createBytesMessage();
        bytes.setStringProperty(WMQConstants.JMS_IBM_FORMAT, getHeaderFormat());
        bytes.setIntProperty(WMQConstants.JMS_IBM_ENCODING, getHeaderEncoding());
        bytes.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET,
                             myProducer.getCCSID());
        bytes.writeBytes(String.format("%1$-" + RECORD_STRUCT_ID_LENGTH + "."
                         + RECORD_STRUCT_ID_LENGTH + "s", getStructID())
                         .getBytes(getMessageCharset()), 0, RECORD_STRUCT_ID_LENGTH);
        bytes.writeInt(getVersion());
        bytes.writeInt(getStructLength());
        return bytes;
    }

    public void setHeaderCharset(String charset) {
        this.headerCharset = charset; }
    public void setHeaderEncoding(int encoding) {
        this.headerEncoding = encoding; }
    public void setHeaderFormat(String headerFormat) {
        this.headerFormat = headerFormat; }
    public void setStructID(String structID) {
        this.structID = structID; }
    public void setStructLength(int structLength) {
        this.structLength = structLength; }
    public void setVersion(int version) {
        this.version = version; }
}
```

Şekil 158. RECORD

```
package com.ibm.mq.id;
import java.io.IOException;
import javax.jms.BytesMessage;
import javax.jms.JMSException;
import com.ibm.mq.headers.MQDataException;

public class MyRecord extends RECORD {
    private static final long serialVersionUID = -370551723162299429L;
    private final static int FLAGS = 1;
    private final static String STRUCT_ID = "MYRD";
    private final static int DATA_LENGTH = 32;
    private final static String FORMAT = "MYRECORD";
    private int flags = FLAGS;
    private String recordData = "ABCDEFGHIJKLMNOPQRSTUVWXYZ012345";

    public MyRecord() {
        super();
        super.setStructID(STRUCT_ID);
        super.setHeaderFormat(FORMAT);
        super.setStructLength(super.getStructLength() + MQLONG_LENGTH
                + DATA_LENGTH);
    }

    public MyRecord(BytesMessage message) throws JMSException, IOException,
            MQDataException {
        super(message);
        setFlags(message.readInt());
        byte[] recordData = new byte[DATA_LENGTH];
        message.readBytes(recordData, DATA_LENGTH);
        setRecordData(new String(recordData, super.getMessageCharset()));
    }

    public static MyRecord get(MyConsumer myConsumer) throws JMSException,
            MQDataException, IOException {
        BytesMessage message = (BytesMessage) myConsumer.receive();
        return new MyRecord(message);
    }

    public int getFlags() { return flags; }
    public String getRecordData() { return recordData; }Â .


    public BytesMessage put(MyProducer myProducer) throws JMSException,
            IOException {
        BytesMessage bytes = super.put(myProducer);
        bytes.writeInt(getFlags());
        bytes.writeBytes(String.format("%1$-" + DATA_LENGTH + "."
                        + DATA_LENGTH + "s",getRecordData())
                        .getBytes(super.getMessageCharset()), 0, DATA_LENGTH);
        myProducer.send(bytes);
        return bytes;
    }

    public void setFlags(int flags) {
        this.flags = flags; }
    public void setRecordData(String recordData) {
        this.recordData = recordData; }
}
```

Şekil 159. MyRecord

```
package com.ibm.mq.id;
import java.io.UnsupportedEncodingException;
import javax.jms.Connection;
import javax.jms.ConnectionFactory;
import javax.jms.Destination;
import javax.jms.JMSException;
import javax.jms.Session;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import com.ibm.mq.headers.CCSID;
import com.ibm.mq.jms.MQDestination;
import com.ibm.msg.client.wmq.WMQConstants;
public abstract class EndPoint {
    public Context ctx;
    public ConnectionFactory cf;
    public Connection connection;
    public Destination destination;
    public Session session;
    protected EndPoint() throws NamingException, JMSException {
        System.setProperty("java.naming.provider.url", "file:/C:/JNDI-Directory");
        System.setProperty("java.naming.factory.initial",
                           "com.sun.jndi.fscontext.RefFSContextFactory");
        ctx = new InitialContext();
        cf = (ConnectionFactory) ctx.lookup("QM1");
        connection = cf.createConnection();
        destination = (Destination) ctx.lookup("Q1");
        ((MQDestination)destination).setReceiveConversion
                                    (WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
        session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE); }
    protected EndPoint(String cFactory, String dest) throws NamingException,
            JMSException {
        System.setProperty("java.naming.provider.url", "file:/C:/JNDI-Directory");
        System.setProperty("java.naming.factory.initial",
                           "com.sun.jndi.fscontext.RefFSContextFactory");
        ctx = new InitialContext();
        cf = (ConnectionFactory) ctx.lookup(cFactory);
        connection = cf.createConnection();
        destination = (Destination) ctx.lookup(dest);
        ((MQDestination)destination).setReceiveConversion
                                    (WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR);
        session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);  }
    public int getCCSID() throws JMSException {
        return (((MQDestination) destination)
                .getIntProperty(WMQConstants.WMQ_CCSID));  }
    public String getCharset() throws UnsupportedEncodingException,
            JMSException {
        return CCSID.getCodepage(getCCSID());  }
    public int getEncoding() throws JMSException {
        return (((MQDestination) destination)
                .getIntProperty(WMQConstants.WMQ_ENCODING));  }
    public boolean getMQDest() throws JMSException {
        if ((((MQDestination) destination).getMessageBodyStyle()
               == WMQConstants.WMQ_MESSAGE_BODY_MQ)
            || ((((MQDestination) destination).getMessageBodyStyle()
                  == WMQConstants.WMQ_MESSAGE_BODY_UNSPECIFIED)
               && (((MQDestination) destination).getTargetClient()
                   == WMQConstants.WMQ_TARGET_DEST_MQ)))
            return true;
        else
            return false; }
    public void setCCSID(int ccsid) throws JMSException {
        ((MQDestination) destination).setIntProperty(WMQConstants.WMQ_CCSID,
                ccsid); }
    public void setEncoding(int encoding) throws JMSException {
        ((MQDestination) destination).setIntProperty(WMQConstants.WMQ_ENCODING,
                encoding); }
    public void setMQBody() throws JMSException {
        ((MQDestination) destination)
                    .setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_UNSPECIFIED);  }
    public void setMQBody(boolean mqbody) throws JMSException {
        if (mqbody)  ((MQDestination) destination)
                    .setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_MQ);
        else         ((MQDestination) destination)
                    .setMessageBodyStyle(WMQConstants.WMQ_MESSAGE_BODY_JMS);  }
    public void setMQClient(boolean mqclient) throws JMSException {
       if (mqclient){
            ((MQDestination) destination).setTargetClient(WMQConstants.WMQ_TARGET_DEST_MQ);
            if (!getMQDest()) setMQBody();
       }
       else
            ((MQDestination) destination).setTargetClient(WMQConstants.WMQ_TARGET_DEST_JMS); }
    }
```

Şekil 160. EndPoint

```
package com.ibm.mq.id;
import javax.jms.JMSException;
import javax.jms.Message;
import javax.jms.MessageProducer;
import javax.naming.NamingException;
public class MyProducer extends EndPoint {
    public MessageProducer producer;
    public MyProducer() throws NamingException, JMSException {
        super();
        producer = session.createProducer(destination); }
    public MyProducer(String cFactory, String dest) throws NamingException,
            JMSException {
        super(cFactory, dest);
        producer = session.createProducer(destination); }
    public void send(Message message) throws JMSException {
        producer.send(message); }
}
```

Şekil 161. MyProducer

```
package com.ibm.mq.id;
import javax.jms.JMSException;
import javax.jms.Message;
import javax.jms.MessageConsumer;
import javax.naming.NamingException;
public class MyConsumer extends EndPoint {
    public MessageConsumer consumer;
    public MyConsumer() throws NamingException, JMSException {
        super();
        consumer = session.createConsumer(destination);
        connection.start();  }
    public MyConsumer(String cFactory, String dest) throws NamingException,
            JMSException {
        super(cFactory, dest);
        consumer = session.createConsumer(destination);
        connection.start();  }
    public Message receive() throws JMSException {
        return consumer.receive();  }
}
```

Şekil 162. MyConsumer

JMS uygulaması için bir WebSphere MQ sınıflarında bağlantı üreticileri ve hedefleri oluşturma ve yapılandırma

A WebSphere MQ classes for JMS application can create connection factories and destinations by retrieving them as administered objects from a Java Naming and Directory Interface (JNDI) namespace, by using the IBM JMS extensions, or by using the WebSphere MQ JMS extensions. Bir uygulama, bağlantı fabrikalarının ve hedeflerin özelliklerini ayarlamak için IBM JMS uzantılarını ya da WebSphere MQ JMS uzantılarını da kullanabilir.

Bağlantı fabrikaları ve hedefler, bir JMS uygulamasının mantık akışındaki noktaları başlatıyor. Bir uygulama, ileti alışverişi sunucusuyla bağlantı yaratmak için ConnectionFactory nesnesini kullanır ve iletilerin gönderileceği hedef ya da ileti göndermek için hedef olarak bir Kuyruk ya da Konu nesnesi kullanır. Bu nedenle, uygulamanın en az bir bağlantı üreticisi ve bir ya da daha çok hedef oluşturması gerekir. Bir bağlantı üreticisi ya da hedefi yarattıktan sonra, uygulamanın bir ya da daha çok özelliğini ayarlayarak nesnenin konfigürasyonunu tanımlamanız gerekebilir.

Özet olarak, bir uygulama bağlantı fabrikalarını ve hedeflerini aşağıdaki şekillerde oluşturabilir ve yapılandırabilir:

Yönetilen nesneleri almak için JNDI kullanılıyor
: Bir yönetici, bir JNDI ad alanında yönetilen nesneler olarak bağlantı üreticileri ve hedefleri oluşturmak ve yapılandırmak için WebSphere MQ JMS yönetim aracını ya da WebSphere MQ Explorer olanağını kullanabilir. Daha sonra, uygulama, yönetilen nesneleri JNDI ad alanından alabilir. Having retrieved an administered object, the application can, if required, set or change one or more of its properties by using either the IBM JMS extensions or the WebSphere MQ JMS extensions.

IBM JMS uzantılarını kullanma
: Uygulama, çalıştırma zamanında dinamik olarak bağlantı üreticileri ve hedefler oluşturmak için IBM JMS uzantılarını kullanabilir. Uygulama önce bir JmsFactoryFactory nesnesi yaratır ve sonra bağlantı

üreticileri ve hedefler yaratmak için bu nesnenin yöntemlerini kullanır. Bir bağlantı üreticisi ya da hedefi yarattıktan sonra, uygulama, özelliklerini ayarlamak için JmsPropertybağlam arabiriminden edinilen yöntemleri kullanabilir. Diğer bir seçenek olarak, uygulama hedefi oluştururken bir hedef için bir ya da daha fazla özellik belirtmek için bir URL ' yi (uniform resource identifier; URI) kullanabilir.

WebSphere MQ JMS uzantılarını kullanma
Uygulama, yürütme sırasında devingen olarak bağlantı üreticileri ve hedefler yaratmak için WebSphere MQ JMS uzantılarını da kullanabilir. Uygulama, bağlantı fabrikaları ve hedefler oluşturmak için sağlanan oluşturucuları kullanır. Bir bağlantı üreticisi ya da hedefi yarattıktan sonra, uygulama, nesnenin özelliklerini ayarlamak için nesne yöntemlerini kullanabilir. Diğer bir seçenek olarak, uygulama hedefi yaratırken bir hedefe ilişkin bir ya da daha çok özellik belirtmek için bir URI kullanabilir.

Bir JMS uygulamasında yönetilen nesneleri almak için JNDI kullanılıyor

Yönetilen nesneleri bir JNDI (Java Naming and Directory Interface; Java Adlandırma ve Dizin Arabirimi) ad alanından almak için, bir JMS uygulamasının ilk bağlamı yaratması ve daha sonra, nesneleri almak için lookup () yöntemini kullanması gerekir.

Bir uygulamanın yönetilen nesneleri JNDI ad alanından alabilmesi için, denetimcinin önce yönetilen nesneleri yaratması gerekir. The administrator can use the WebSphere MQ JMS administration tool or WebSphere MQ Explorer to create and maintain administered objects in a JNDI namespace. WebSphere MQ JMS yönetim aracını nasıl kullanabilmeye ilişkin bilgi için bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896. WebSphere MQ Explorer olanağının nasıl kullanılacağı hakkında bilgi için WebSphere MQ Explorer ile sağlanan yardım konusuna bakın. Ancak, bir uygulama sunucusu, nesneleri oluşturmak ve korumak için yönetilen nesneler ve kendi araçları için kendi havuzunu sağlar.

Bir JNDI ad alanından denetlenen nesneleri almak için, aşağıdaki örnekte gösterildiği gibi, bir uygulamanın öncelikle başlangıç bağlamı yaratması gerekir:

```
import javax.jms.*;
import javax.naming.*;
import javax.naming.directory.*;
.
.
.
String url = "ldap://server.company.com/o=company_us,c=us";
String icf = "com.sun.jndi.ldap.LdapCtxFactory";
.
java.util.Hashtable environment = new java.util.Hashtable();
environment.put(Context.PROVIDER_URL, url);
environment.put(Context.INITIAL_CONTEXT_FACTORY, icf);
Context ctx = new InitialDirContext(environment);
```

Bu kodda, `url` ve `icf` dizgi değişkenleri aşağıdaki anlamlara sahiptir:

url
Dizin hizmetinin URL (uniform resource locator; URL adresi). URL adresi aşağıdaki biçimlerden birine sahip olabilir:

- `ldap://`*`hostname`*`/`*`contextName`*, for a directory service based on an LDAP server
- `file:/`*`directoryPath`*, yerel dosya sistemine dayalı bir dizin hizmeti için

iCF
İlk bağlam üreticisinin sınıf adı; aşağıdaki değerlerden biri olabilir:

- `com.sun.jndi.ldap.LdapCtxFactory`, for a directory service based on an LDAP server
- `com.sun.jndi.fscontext.RefFSContextFactory`, yerel dosya sistemine dayalı bir dizin hizmeti için

Bir JNDI paketi ve bir LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) hizmet sağlayıcısının bazı birleşimlerinin, LDAP hata 84 'ünün oluşmasına neden olduğunu unutmayın. To resolve this problem, insert the following line of code before the call to InitialDirContext():

```
environment.put(Context.REFERRAL, "throw");
```

İlk bağlam elde edildikten sonra, uygulama, aşağıdaki örnekte gösterildiği gibi, lookup () yöntemini kullanarak, JNDI ad alanından denetimli nesneleri alabilir:

```
ConnectionFactory factory;
Queue queue;
Topic topic;
.
.
.
factory = (ConnectionFactory)ctx.lookup("cn=myCF");
queue = (Queue)ctx.lookup("cn=myQ");
topic = (Topic)ctx.lookup("cn=myT");
```

Bu kod, LDAP tabanlı bir ad alanından aşağıdaki nesneleri alır:

- myCFadına bağlanan bir ConnectionFactory nesnesi
- myQadına bağlı bir Kuyruk nesnesi
- myTadına bağlı bir Konu nesnesi

IBM JMS uzantılarını kullanma

JMS içinWebSphere MQ sınıfları, IBM JMS uzantıları olarak adlandırılan JMS API ' ya bir uzantı kümesi içerir. Uygulama, yürütme sırasında dinamik olarak bağlantı fabrikaları ve hedefler yaratmak ve JMS nesneleri için WebSphere MQ sınıflarının özelliklerini ayarlamak için bu uzantıları kullanabilir. Uzantılar, herhangi bir ileti alışverişi sağlayıcısıyla birlikte kullanılabilir.

IBM JMS uzantıları, aşağıdaki paketlerdeki bir arabirim ve sınıflardan oluşan bir kümedir:

- com.ibm.msg.client.jms
- com.ibm.msg.client.services

Paketler, `<MQ_Install_Dir>/java/lib`biçinde yer alan `com.ibm.mqjms.jar` içinde bulunabilir.

Bu uzantılar aşağıdaki işlevi sağlar:

- Bir Java Adlandırma ve Dizin Arabirimi (JNDI) ad alanından yönetilen nesneler olarak almak yerine, yürütme sırasında dinamik olarak bağlantı fabrikaları ve hedefler oluşturmak için kullanılan bir fabrikada kullanılan bir mekanizma.
- JMS nesneleri için WebSphere MQ sınıflarının özelliklerini ayarlamak için kullanılan yöntemler kümesi
- Bir sorunla ilgili ayrıntılı bilgi edinme yöntemleriyle ilgili kural dışı durum sınıfları kümesi
- İzlemeyi denetlemeye ilişkin yöntemler kümesi
- JMS için WebSphere MQ sınıflarıyla ilgili sürüm bilgilerini edinmeye yönelik bir dizi yöntem

Çalıştırma zamanında dinamik olarak bağlantı fabrikaları ve varış noktaları oluşturma ve bunların özelliklerini alma ve alma konusunda IBM JMS uzantıları, WebSphere MQ JMS uzantıları için alternatif bir arabirim kümesi sağlar. Ancak, WebSphere MQ JMS uzantıları WebSphere MQ ileti alışverişi sağlayıcısına özgüyken, IBM JMS uzantıları WebSphere MQ için özel değildir ve “Katmanlı bir mimari” sayfa 766içinde açıklanan katmanlı mimari içinde herhangi bir ileti alışverişi sağlayıcısıyla birlikte kullanılabilir.

The interface com.ibm.msg.client.wmq.WMQConstants contains the definitions of constants, which an application can use when setting the properties of WebSphere MQ classes for JMS objects using the IBM JMS extensions. Arabirim, herhangi bir ileti alışverişi sağlayıcısından bağımsız olarak WebSphere MQ ileti alışverişi sağlayıcısı ve JMS değişmezleri için sabit değerler içerir.

Aşağıdaki içe aktarma deyimlerinin çalıştırıldığı varsayılarak, aşağıdaki kod örnekleri verilmiştir:

```
import com.ibm.msg.client.jms.*;
import com.ibm.msg.client.services.*;
import com.ibm.msg.client.wmq.WMQConstants;
```

Bağlantı üreticileri ve hedefleri oluşturma

Bir uygulamanın IBM JMS uzantılarını kullanarak bağlantı üreticileri ve hedefleri oluşturabilmesi için önce bir JmsFactoryFactory nesnesi yaratması gerekir. Bir JmsFactoryFactory nesnesi yaratmak için, uygulama JmsFactoryFactory sınıfının getInstance() yöntemini çağırır. Aşağıdaki örnekte gösterildiği gibi:

```
JmsFactoryFactory ff = JmsFactoryFactory.getInstance(JmsConstants.WMQ_PROVIDER);
```

getInstance() çağrısının üzerindeki değiştirge, seçilen ileti alışverişi sağlayıcısı olarak WebSphere MQ ileti alışverişi sağlayıcısını tanıtan bir sabittir. Daha sonra uygulama, bağlantı fabrikaları ve hedefler yaratmak için JmsFactoryFactory nesnesini kullanabilir.

Bir bağlantı üreticisi yaratmak için, uygulama JmsFactoryFactory nesnesinin createConnectionFactory () yöntemini çağırır. Aşağıdaki örnekte gösterildiği gibi:

```
JmsConnectionFactory factory = ff.createConnectionFactory();
```

Bu deyim, tüm özellikleri için varsayılan değerlere sahip bir JmsConnectionFactory nesnesi yaratır; bu, uygulamanın bağ tanımları kipindeki varsayılan kuyruk yöneticisine bağlandığı anlamına gelir. Bir uygulamanın istemci kipinde bağlanmasını istiyorsanız ya da varsayılan kuyruk yöneticisi dışında bir kuyruk yöneticisine bağlanmak istiyorsanız, bu bağlantıyı yaratmadan önce uygulamanın JmsConnectionFactory nesnesinin uygun özelliklerini ayarlaması gerekir. Bunun nasıl yapacagına ilişkin bilgi için bkz. “JMS nesneleri için WebSphere MQ sınıflarının özelliklerinin ayarlanması” sayfa 833.

JmsFactoryFactory sınıfı, aşağıdaki tiplere ilişkin bağlantı üreticileri yaratmak için de yöntemler içerir:

- JmsQueueConnectionFactory
- JmsTopicConnectionFactory
- JmsXAConnectionÜreticisi
- JmsXAQueueConnectionFactory
- JmsXATopicConnectionFactory

Bir kuyruk nesnesi yaratmak için, uygulama JmsFactoryFactory nesnesinin createQueue() yöntemini çağırır. Aşağıdaki örnekte gösterildiği gibi:

```
JmsQueue q1 = ff.createQueue("Q1");
```

Bu deyim, tüm özellikleri için varsayılan değerleri içeren bir JmsQueue nesnesi yaratır. Nesne, yerel kuyruk yöneticisine ait Q1 adlı bir WebSphere MQ kuyruğunu temsil eder. Bu kuyruk, yerel bir kuyruk, bir diğer ad kuyruğu ya da uzak kuyruk tanımlaması olabilir.

createQueue() yöntemi, bir parametre olarak bir kuyruk tekstili kaynak tanıtıcısını (URI) da kabul edebilir. Kuyruk URI 'si, bir WebSphere MQ kuyruğunun adını ve isteğe bağlı olarak, kuyruğa sahip olan kuyruk yöneticisinin adını ve JmsQueue nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki deyimde bir kuyruk URI örneği yer almaktadır:

```
JmsQueue q2 = ff.createQueue("queue://QM2/Q2?persistence=2&priority=5");
```

The JmsQueue object created by this statement represents a WebSphere MQ queue called Q2 that is owned by queue manager QM2, and all messages sent to this destination are persistent and have a priority of 5. Kuyruk URI ' leri hakkında daha fazla bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844. Bir JmsQueue nesnesinin özelliklerini ayara alternatif bir yöntem için bkz. “JMS nesneleri için WebSphere MQ sınıflarının özelliklerinin ayarlanması” sayfa 833.

Bir Konu nesnesi oluşturmak için bir uygulama, aşağıdaki örnekte gösterildiği gibi JmsFactoryFactory nesnesinin createTopic() yöntemini kullanabilir:

```
JmsTopic t1 = ff.createTopic("Sport/Football/Results");
```

Bu deyim, tüm özellikleri için varsayılan değerleri içeren bir JmsTopic nesnesi yaratır. Nesne, Sport/Football/Results adlı bir konuyu gösterir.

createTopic() yöntemi bir konu URI 'sini parametre olarak da kabul edebilir. Konu URI 'si, bir konunun adını ve isteğe bağlı olarak, JmsTopic nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki deyimler bir konu URI örneğini içerir:

```
String s1 = "topic://Sport/Tennis/Results?persistence=1&priority=0";
JmsTopic t2 = ff.createTopic(s1);
```

Bu deyimler tarafından yaratılan JmsTopic nesnesi, Sport/Tennis/Results adlı bir konuyu gösterir ve bu hedefe gönderilen tüm iletiler kalıcı değildir ve 0 önceliğine sahiptir. Konu URI ' leri hakkında daha fazla bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844. Bir JmsTopic nesnesinin özelliklerini ayara alternatif bir yöntem için bkz. “JMS nesneleri için WebSphere MQ sınıflarının özelliklerinin ayarlanması” sayfa 833.

Bir uygulama, bir bağlantı üreticisi ya da hedefi yarattıktan sonra, bu nesne yalnızca seçilen ileti sistemi sağlayıcısıyla kullanılabilir.

JMS nesneleri için WebSphere MQ sınıflarının özelliklerinin ayarlanması

To set the properties of WebSphere MQ classes for JMS objects using the IBM JMS extensions, an application uses the methods of the com.ibm.msg.client.JmsPropertyContext interface.

Her Java veri tipi için, JmsPropertyBağlam arabirimi, o veri tipine sahip bir özelliğin değerini ayarlamak için bir yöntem ve bu veri tipine sahip bir özelliğin değerini almak için bir yöntem içerir. Örneğin, bir uygulama setIntProperty () yöntemini, bir tamsayı değeri olan bir özelliği ayarlamayı çağırır ve getIntProperty () yöntemini, bir tamsayı değeri olan bir özelliği almak için çağırır.

com.ibm.mq.jms paketindeki sınıfların eşgörünümleri de JmsPropertybağlam arabiriminin yöntemlerini devralır. Bu nedenle, bir uygulama bu yöntemleri kullanarak MQConnectionFactory, MQQueue ve MQTopic nesnelerine ilişkin özellikleri ayarlar.

Bir uygulama JMS nesnesi için bir WebSphere MQ sınıfları yarattığında, varsayılan değerleri olan özellikler otomatik olarak ayarlanır. Bir uygulama bir özelliği ayarladığında, yeni değer özelliğin sahip olduğu önceki değerin yerini alır. Bir özellik ayarlandıktan sonra silinemez, ancak değeri değiştirilebilir.

Bir uygulama, bir özelliği özellik için geçerli olmayan bir değere ayarlamaya çalışırsa, JMS için WebSphere MQ classicas bir JMSException kural dışı durumunu atar. Bir uygulama ayarlanmamış bir özelliği alma girişiminde bulunursa, davranış JMS belirtiminde anlatıldığı gibi olur. WebSphere MQ class for JMS, ilkel veri tipleri için bir NumberFormatkural dışı durum kural dışı durumu yayınladı ve gönderme yapılan veri tipleri için boş değer döndürür.

Bir uygulama, JMS nesnesi için bir WebSphere MQ sınıflarının önceden tanımlanmış özelliklerine ek olarak, kendi özelliklerini ayarlayabilir. Bu uygulama tanımlı özellikler, JMS için WebSphere MQ sınıfları tarafından yok sayıldı.

JMS nesnelerine ilişkin WebSphere MQ sınıflarının özellikleri hakkında daha fazla bilgi için IBM WebSphere MQ classes for JMS nesnelerinin özellikleribaşlıklı konuya bakın.

The following code is an example of how to set properties using the IBM JMS extensions. Kod, bir bağlantı üreticisinin beş özelliğini ayarlar.

```
factory.setIntProperty(WMQConstants.WMQ_CONNECTION_MODE,
                       WMQConstants.WMQ_CM_CLIENT);
factory.setStringProperty(WMQConstants.WMQ_QUEUE_MANAGER, "QM1");
factory.setStringProperty(WMQConstants.WMQ_HOST_NAME, "HOST1");
factory.setIntProperty(WMQConstants.WMQ_PORT, 1415);
factory.setStringProperty(WMQConstants.WMQ_CHANNEL, "QM1.SVR");
factory.setStringProperty(WMQConstants.WMQ_APPLICATIONNAME, "My Application");
```

The effect of setting these properties is that the application connects to queue manager QM1 in client mode, using an MQI channel called QM1.SVR. Kuyruk yöneticisi, anasistem adı HOST1olan bir sistemde çalışıyor ve kuyruk yöneticisinin dinleyicisi 1415 numaralı kapıda dinlemede. Bu bağlantı ve bunun altındaki oturumlarla ilişkili diğer kuyruk yöneticisi bağlantıları, onlarla ilişkili uygulama adına "Uygulamam" adını verdi.

Not: z/OS platformlarında çalışan kuyruk yöneticileri uygulama adlarını ayarlamamaktadır ve bu nedenle bu ayar yoksayılır.

The JmsPropertyContext interface also contains the setObjectProperty() method, which an application can use to set properties. Yöntemin ikinci parametresi, özelliğin değerini sarmalayan bir nesnedir. Örneğin, aşağıdaki kod, 1415 değerini soyan bir Tamsayı nesnesi yaratır ve daha sonra, bağlantı üreticisinin PORT özelliğini 1415 değerine ayarlamak için setObjectProperty () çağrılarını çağırır:

```
Integer port = new Integer(1415);
factory.setObjectProperty(WMQConstants.WMQ_PORT, port);
```

Bu nedenle, bu kod aşağıdaki deyime eşdeğerdir:

```
factory.setIntProperty(WMQConstants.WMQ_PORT, 1415);
```

Tersi durumda, getObjectProperty () yöntemi, bir özelliğin değerini sarmalayan bir nesneyi döndürür.

Bir özellik değerinin bir veri tipinden diğerine örtük olarak dönüştürülmesi

Bir uygulama, JMS nesnesi için bir WebSphere MQ sınıflarının özelliğini ayarlamak ya da almak için JmsPropertyBağlamı arabirimi yöntemini kullandığında, özelliğin değeri bir veri tipinden diğerine örtük olarak dönüştürülebiliyor.

Örneğin, aşağıdaki deyim JmsQueue nesnesinin PRIORITY özelliğini ayarlar. q1:

```
q1.setStringProperty(WMQConstants.WMQ_PRIORITY, "5");
```

PRIORITY özelliğinin bir tamsayı değeri vardır; dolayısıyla, setStringProperty () çağrısı örtük olarak "5" (kaynak değer) dizgisini (kaynak değer) tamsayı 5 'e (hedef değer) dönüştürür ve daha sonra, PRIORITY özelliğinin değeri olur.

Bunun tersine, aşağıdaki deyim JmsQueue nesnesinin PRIORITY özelliğini alır. q1:

```
String s1 = q1.getStringProperty(WMQConstants.WMQ_PRIORITY);
```

PRIORITY özelliğinin değeri olan 5 numaralı tamsayı (kaynak değer), getStringProperty () çağrısıyla örtük olarak "5" dizgisine (hedef değer) dönüştürüldü.

JMS için WebSphere MQ sınıfları tarafından desteklenen dönüştürmeler Çizelge 122 sayfa 834içinde gösterilir.

Çizelge 122. Bir veri tipinden diğerine dönüştürme desteklenir

Kaynak veri tipi	**Desteklenen hedef veri tipleri**
boole	Dizgi
Byte	int, long, short, String
DAMGA	Dizgi
çift	Dizgi
kayan nokta	çift, Dizgi
int	uzun, Dizgi
uzun	Dizgi
kısa	int, long, String
Dizgi	boolean, byte, double, float, int, long, short

Desteklenen dönüştürmeleri yöneten genel kurallar aşağıdaki gibidir:

- Sayısal değerler, dönüştürme sırasında bir veri tipinden diğerine dönüştürülebilecek şekilde dönüştürülebilirler. For example, a value with data type `int` can be converted into a value with data type `long`, but cannot be converted into a value with data type `short`.
- Herhangi bir veri tipindeki bir değer bir dizgiye dönüştürülebiliyor.
- Dizgi, dönüştürme için doğru biçimde olması koşuluyla, başka bir veri tipindeki (`char`dışında) bir değere dönüştürülebilmektedir. Bir uygulama, doğru biçimde olmayan bir dizgiyi dönüştürmeyi denerse, JMS için WebSphere MQ sınıfları NumberFormatkural dışı durumu kural dışı durumuna döndürür.
- Bir uygulama, desteklenmeyen bir dönüştürme girişiminde bulunursa, JMS için WebSphere MQ sınıfları bir MessageFormatkural dışı durumu kural dışı durumuna döndürür.

Bir veri tipinden başka bir veri tipine dönüştürme için belirli kurallar şunlardır:

- Bir boole değerini dizgiye dönüştürürken, `true` değeri "true"dizgisine dönüştürülür ve `false` değeri "false"dizgisine dönüştürülür.
- Bir dizgiyi bir boole değerine dönüştürürken, "doğru" dizgisi (büyük-küçük harfe duyarlı değil) `true`değerine dönüştürülür ve "yanlış" dizgisi (büyük/küçük harfe duyarlı değildir) `false`değerine dönüştürülür. Diğer herhangi bir dizgi `false`' a dönüştürülür.
- Bir dizeyi `byte`, `int`, `long`ya da `short`veri tipli bir değere dönüştürürken, dizginin şu biçimde olması gerekir:

  [*boşluk*] [*sign*]*haneler*

  Dizginin bileşenlerinin anlamları aşağıdaki gibidir:

  boşluklar
  İsteğe bağlı olarak baştaki boş karakterler.

  İşaret
  İsteğe bağlı artı işareti (+) ya da eksi işareti (-).

  Rakamlar
  Bitişik basamaklar dizisi (0-9). En az bir basamak var olmalıdır.

  Basamaklar sırasından sonra, dizgi sayı olmayan diğer karakterleri içerebilir, ancak bu karakterlere ilk girildiği anda dönüştürme durakları durur. Dizilimin ondalık bir tamsayıyı temsil ettiği varsayılır.

  Dizgi doğru biçimde değilse, JMS için WebSphere MQ sınıfları NumberFormatkural dışı durumu kural dışı durumunu atar.
- Bir dizgiyi `double` ya da `float`veri tipine sahip bir değere dönüştürürken, dizginin şu biçimde olması gerekir:

  [*boşluk*] [*işaret*]*basamaklar*[*e_char*[*e_işareti*]*e_basamakları*]

  Dizginin bileşenlerinin anlamları aşağıdaki gibidir:

  boşluklar
  İsteğe bağlı olarak baştaki boş karakterler.

  İşaret
  İsteğe bağlı artı işareti (+) ya da eksi işareti (-).

  Rakamlar
  Bitişik basamaklar dizisi (0-9). En az bir basamak var olmalıdır.

  e_char
  E ya da *E*olan bir üstel karakter.

  e_sign
  Üstel için isteğe bağlı artı işareti (+) ya da eksi işareti (-).

  e_basamakları
  Üs için bitişik basamaklar dizisi (0-9). Dizgi bir üstel karakter içeriyorsa en az bir basamak var olmalıdır.

Basamaklar sırasından sonra ya da bir üssü temsil eden isteğe bağlı karakterler, dizgi olmayan diğer karakterleri içerebilir, ancak bu karakterlere ilk girişe ulaşıldığında dönüştürme durakları durur. Dizginin, 10 'un gücü olan bir üstel bir ondalık kayan noktalı sayıyı temsil ettiği varsayılır.

Dizgi doğru biçimde değilse, JMS için WebSphere MQ sınıfları NumberFormatkural dışı durumu kural dışı durumunu atar.

- Bir sayısal değeri (veri tipi byteolan bir değer dahil) bir dizgiye dönüştürürken, değer, o değere ilişkin ASCII karakterini içeren dizeyi değil, değerin ondalık sayı olarak dizgi gösterimine dönüştürülür. For example, the integer 65 is converted to the string "65", not the string "A".

Tek bir çağrıda birden çok özellik ayarlama

The JmsPropertyContext interface also contains the setBatchProperties() method, which an application can use to set more than one property in a single call. Yöntemin değiştirgesi, özellik ad-değer çiftleri kümesini sarmalayan bir Eşlem nesnesidir.

Örneğin, aşağıdaki kod, bir bağlantı üreticisinin beş özelliğini “JMS nesneleri için WebSphere MQ sınıflarının özelliklerinin ayarlanması” sayfa 833içinde gösterildiği gibi ayarlamak için setBatchProperties () yöntemini kullanır. Bu kod, Harita arabirimini gerçekleştiren HashMap sınıfının bir eşgörünümünü oluşturur.

```
HashMap batchProperties = new HashMap();
batchProperties.put(WMQConstants.WMQ_CONNECTION_MODE,
                    new Integer(WMQConstants.WMQ_CM_CLIENT));
batchProperties.put(WMQConstants.WMQ_QUEUE_MANAGER, "QM1");
batchProperties.put(WMQConstants.WMQ_WMQ_HOST_NAME, "HOST1");
batchProperties.put(WMQConstants.WMQ_PORT, "1414");
batchProperties.put(WMQConstants.WMQ_CHANNEL, "QM1.SVR");
factory.setBatchProperties(batchProperties);
```

Map.put() yönteminin ikinci parametresinin bir nesne olması gerektiğini unutmayın. Bu nedenle, ilkel bir veri tipine sahip bir özellik değeri, bir nesne içinde ya da örnekte gösterildiği gibi bir dizgi tarafından gösterilmelidir.

setBatchProperties () yöntemi her özelliği doğrular. setBatchProperties () yöntemi bir özelliği ayarlayamıyorsa, örneğin değeri geçerli değil, belirtilen özelliklerin hiçbiri ayarlanmamış.

Özellik adları ve değerleri

Bir uygulama, JMS nesneleri için WebSphere MQ sınıflarının özelliklerini ayarlamak ve almak için JmsPropertyContext arabiriminin yöntemlerini kullanıyorsa, uygulama aşağıdaki yöntemlerden birini kullanarak özelliklerin adlarını ve değerlerini belirtebilir. Each of the accompanying examples shows how to set the PRIORITY property of the JmsQueue object q1 so that a message sent to the queue has the priority specified on the send() call.

com.ibm.msg.client.wmq.WMQConstants arabiriminde sabit değerler olarak tanımlanan özellik adlarını ve değerlerini kullanma

Aşağıdaki deyim, bu şekilde özelliklerin adlarını ve değerlerini nasıl belirleyeceğini gösteren bir örnektir:

```
q1.setIntProperty(WMQConstants.WMQ_PRIORITY, WMQConstants.WMQ_PRI_APP);
```

Kuyruk ve konu tekstili kaynak tanıtıcıları (URI ' ler) içinde kullanılabilecek özellik adlarını ve değerlerini kullanma

Aşağıdaki deyim, bu şekilde özelliklerin adlarını ve değerlerini nasıl belirleyeceğini gösteren bir örnektir:

```
q1.setIntProperty("priority", -2);
```

Bu şekilde yalnızca hedeflerin özelliklerinin adları ve değerleri belirtilebilir.

WebSphere MQ JMS yönetim aracı tarafından tanınan özellik adları ve değerleri kullanılıyor
Aşağıdaki deyim, bu şekilde özelliklerin adlarını ve değerlerini nasıl belirleyeceğini gösteren bir örnektir:

```
q1.setStringProperty("PRIORITY", "APP");
```

Özellik adının kısa biçimi de, aşağıdaki deyimde gösterildiği gibi kabul edilebilir:

```
q1.setStringProperty("PRI", "APP");
```

Bir uygulama bir özellik aldığında, döndürülen değer, uygulamanın özelliğin adını belirtme yöntemine bağlıdır. For example, if an application specifies the constant WMQConstants.WMQ_PRIORITY as the property name, the value returned is the integer -2:

```
int n1 = getIntProperty(WMQConstants.WMQ_PRIORITY);
```

Uygulama, özellik adı olarak "priority" dizesini belirtiyorsa, aynı değer döndürülür:

```
int n2 = getIntProperty("priority");
```

Ancak, uygulama özellik adı olarak "PRIMARY" dizgisini ya da "PRI" dizgisini belirtiyorsa, döndürülen değer "APP" dizgisidir:

```
String s1 = getStringProperty("PRI");
```

JMS için içeride, WebSphere MQ sınıfları, özellik adlarını ve değerlerini com.ibm.msg.client.wmq.WMQConstants arabiriminde tanımlı hazır bilgi değerleri olarak saklar. Bu, özellik adları ve değerleri için tanımlanan kurallı biçimdir. Genel bir kural olarak, bir uygulama, özellik adlarını ve değerlerini belirtmenin diğer iki yönteminden birini kullanarak özellikleri ayarlarsa, JMS için WebSphere MQ sınıfları, belirtilen giriş biçimindeki adları ve değerleri kurallı biçime dönüştürmesi gerekir. Benzer şekilde, bir uygulama özellik adlarını ve değerlerini belirtmenin diğer iki yolundan birini kullanarak özellikler alıyorsa, JMS için WebSphere MQ class for JMS, adları belirtilen giriş biçiminden kurallı biçime dönüştürmeli ve değerleri kurallı biçimden gerekli çıkış biçimine dönüştürmelidir. Bu dönüştürmeleri gerçekleştirmek için, bu dönüştürmelerin başarımı olumsuz sonuçlar doğurabilir.

Özellik adları ve kural dışı durumlar tarafından döndürülen değerler, izleme dosyalarında ya da JMS günlüğü için WebSphere MQ sınıflarında her zaman kurallı biçimde olur.

Harita arabirimini kullanma

JmsPropertyBağlam arabirimi, java.util.Map arabirimini genişletir. Bu nedenle, bir uygulama, JMS nesnesi için bir WebSphere MQ sınıflarının özelliklerine erişmek için Map arabiriminin yöntemlerini kullanabilir.

Örneğin, aşağıdaki kodda bir bağlantı üreticisinin tüm özelliklerinin adları ve değerleri yazdırılıyor. Kod, özelliklerin adlarını ve değerlerini almak için yalnızca Harita arabirimi yöntemlerini kullanır.

```
// Get the names of all the properties
Set propNames = factory.keySet();

// Loop round all the property names and get the property values
Iterator iterator = propNames.iterator();
while (iterator.hasNext()){
    String pName = (String)iterator.next();
    System.out.println(pName+"="+factory.get(pName));
}
```

Harita arabirimi yöntemlerinin kullanılması, özellik doğrulamalarını ya da dönüştürmeleri atlamaz.

WebSphere MQ JMS uzantılarını kullanma

JMS için WebSphere MQ sınıfları, WebSphere MQ JMS uzantıları olarak adlandırılan JMS API ' ya yönelik uzantılar içerir. Bir uygulama, yürütme sırasında dinamik olarak bağlantı fabrikaları ve hedefler oluşturmak ve bağlantı fabrikaları ve varış noktalarının özelliklerini ayarlamak için bu uzantıları kullanabilir.

JMS için WebSphere MQ sınıfları, com.ibm.jms ve com.ibm.mq.jmspaketlerinde bir sınıf kümesi içerir. Bu sınıflar JMS arabirimlerini uygular ve WebSphere MQ JMS uzantılarını içerir. Aşağıdaki deyimler tarafından içe aktarılmış olarak bu paketlerin içe aktarıldığı varsayılarak örnek verilebilir:

```
import com.ibm.jms.*;
import com.ibm.mq.jms.*;
```

Bir uygulama, aşağıdaki işlevleri gerçekleştirmek için WebSphere MQ JMS uzantılarını kullanabilir:

- Bir Java Adlandırma ve Dizin Arabirimi (JNDI) ad alanından yönetilen nesneler olarak almak yerine, yürütme sırasında dinamik olarak bağlantı fabrikaları ve hedefler oluşturun
- Bağlantı üreticilerinin ve hedeflerin özelliklerini belirleyin

Bağlantı üreticileri yaratılması

Bir uygulama, bir bağlantı üreticisi yaratmak için aşağıdaki örnekteki gibi MQConnectionFactory oluşturucusunu kullanabilir:

```
MQConnectionFactory factory = new MQConnectionFactory();
```

Bu deyim, tüm özellikleri için varsayılan değerlere sahip bir MQConnectionFactory nesnesi yaratır; bu nesne, uygulamanın bağ tanımları kipinde varsayılan kuyruk yöneticisine bağlandığı anlamına gelir. Bir uygulamanın istemci kipinde bağlanmasını istiyorsanız ya da varsayılan kuyruk yöneticisi dışında bir kuyruk yöneticisine bağlanmak istiyorsanız, bağlantı yaratılmadan önce uygulamanın MQConnectionFactory nesnesinin uygun özelliklerini ayarlaması gerekir. Bunun nasıl yapacağına ilişkin bilgi için bkz. “Bağlantı üreticilerinin özelliklerinin ayarlanması” sayfa 838.

Bir uygulama aşağıdaki tiplerin bağlantı üreticilerinin benzer şekilde yaratılabilir:

- MQQueueConnectionÜreticisi
- MQTopicConnectionÜreticisi
- MQXAConnectionFactory
- MQXAQueueConnectionÜreticisi
- MQXATopicConnectionÜreticisi

Bağlantı üreticilerinin özelliklerinin ayarlanması

Bir uygulama, bağlantı üreticisinin uygun yöntemlerini çağırarak bir bağlantı üreticisinin özelliklerini ayarlayabilir. Bağlantı üreticisi, yönetilen bir nesne ya da yürütme sırasında devingen olarak yaratılmış bir nesne olabilir.

Örneğin, aşağıdaki kodu göz önünde bulundurun:

```
MQConnectionFactory factory = new MQConnectionFactory();
.
factory.setTransportType(WMQConstants.WMQ_CM_CLIENT);
factory.setQueueManager("QM1");
factory.setHostName("HOST1");
factory.setPort(1415);
factory.setChannel("QM1.SVR");
```

Bu kod bir MQConnectionFactory nesnesi yaratır ve nesnenin beş özelliğini ayarlar. The effect of setting these properties is that the application connects to queue manager QM1 in client mode using an MQI channel called QM1.SVR. Kuyruk yöneticisi, anasistem adı HOST1olan bir sistemde çalışıyor ve kuyruk yöneticisinin dinleyicisi 1415 numaralı kapıda dinlemede.

Bir aracıya gerçek zamanlı bağlantı için bir uygulama aşağıdaki kodu kullanabilir:

```
MQConnectionFactory factory = new MQConnectionFactory();
.
factory.setTransportType(WMQConstants.WMQ_CM_DIRECT);
factory.setHostName("HOST2");
factory.setPort(1507);
```

This code assumes that the broker is running on a system with host name HOST2 and listening on port number 1507.

Bir aracıya gerçek zamanlı bağlantı kullanan bir uygulama, ileti alışverişi için yalnızca yayınlama/abone olma biçimlerinden birini kullanabilir. Bu, ileti alışverişi noktadan noktaya iletişim biçesini kullanamaz.

Yalnızca bir bağlantı üreticisinin özelliklerinin belirli birleşimleri geçerlidir. Hangi birleşimlerin geçerli olduğu hakkında bilgi için bkz. JMS nesneleri için WebSphere MQ sınıflarının özellikleri arasındaki bağımlılıklar .

Bir bağlantı üreticisinin özellikleri ve özelliklerini ayarlamak için kullanılan yöntemlere ilişkin ek bilgi için IBM WebSphere MQ classes for JMS nesnelerinin özellikleribaşlıklı konuya bakın.

Hedefler oluşturma

Bir kuyruk nesnesi yaratmak için, bir uygulama MQQueue oluşturucusunu aşağıdaki örnekte gösterildiği gibi kullanabilir:

```
MQQueue q1 = new MQQueue("Q1");
```

Bu deyim, tüm özellikleri için varsayılan değerleri içeren bir MQQueue nesnesi yaratır. Nesne, yerel kuyruk yöneticisine ait Q1 adlı bir WebSphere MQ kuyruğunu temsil eder. Bu kuyruk, yerel bir kuyruk, bir diğer ad kuyruğu ya da uzak kuyruk tanımlaması olabilir.

Aşağıdaki örnekte gösterildiği gibi, MQQueue oluşturucusunun alternatif bir biçiminin iki değiştirgesi vardır:

```
MQQueue q2 = new MQQueue("QM2", "Q2");
```

The MQQueue object created by this statement represents a WebSphere MQ queue called Q2 that is owned by queue manager QM2. Bu şekilde tanımlanan kuyruk yöneticisi, yerel kuyruk yöneticisi ya da uzak kuyruk yöneticisi olabilir. If it is a remote queue manager, WebSphere MQ must be configured so that, when the application sends a message to this destination, Websphere MQ can route the message from the local queue manager to the remote queue manager.

MQQueue oluşturucusu, tek bir değiştirge olarak bir kuyruk tipi kaynak tanıtıcısını (URI) da kabul edebilir. Kuyruk URI 'si, bir WebSphere MQ kuyruğunun adını ve isteğe bağlı olarak, kuyruğun sahibi olan kuyruk yöneticisinin adını ve MQQueue nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki deyimde bir kuyruk URI örneği yer almaktadır:

```
MQQueue q3 = new MQQueue("queue://QM3/Q3?persistence=2&priority=5");
```

The MQQueue object created by this statement represents a WebSphere MQ queue called Q3 that is owned by queue manager QM3, and all messages sent to this destination are persistent and have a priority of 5. Kuyruk URI ' leri hakkında daha fazla bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844. Bir MQQueue nesnesinin özelliklerini ayara alternatif bir yol için bkz. “Hedeflerin özelliklerinin ayarlanması” sayfa 840.

Bir Konu nesnesi yaratmak için, bir uygulama MQTopic oluşturucusunu aşağıdaki örnekte gösterildiği gibi kullanabilir:

```
MQTopic t1 = new MQTopic("Sport/Football/Results");
```

Bu deyim, tüm özellikleri için varsayılan değerleri içeren bir MQTopic nesnesi yaratır. Nesne, Sport/Football/Results adlı bir konuyu gösterir.

MQTopic oluşturucusu bir konu URI 'sini değiştirge olarak da kabul edebilir. Konu URI 'si, bir konunun adını ve isteğe bağlı olarak MQTopic nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki deyimde bir konu URI 'si örneği yer almaktadır:

```
MQTopic t2 = new MQTopic("topic://Sport/Tennis/Results?persistence=1&priority=0");
```

Bu deyim tarafından yaratılan MQTopic nesnesi, Sport/Tennis/Results adlı bir konuyu gösterir ve bu hedefe gönderilen tüm iletiler kalıcı değildir ve 0 önceliğine sahiptir. Konu URI ' leri hakkında daha fazla

bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844. Bir MQTopic nesnesinin özelliklerini ayara alternatif bir yol için bkz. “Hedeflerin özelliklerinin ayarlanması” sayfa 840.

Hedeflerin özelliklerinin ayarlanması

Bir uygulama, hedefe ilişkin uygun yöntemleri çağırarak, bir hedefin özelliklerini ayarlayabilir. Hedef, yönetilen bir nesne ya da yürütme sırasında dinamik olarak yaratılan bir nesne olabilir.

Örneğin, aşağıdaki kodu göz önünde bulundurun:

```
MQQueue q1 = new MQQueue("Q1");
.
q1.setPersistence(WMQConstants.WMQ_PER_PER);
q1.setPriority(5);
```

Bu kod bir MQQueue nesnesi yaratır ve nesnenin iki özelliğini ayarlar. Bu özellikleri ayarlamaya ilişkin etki, hedefe gönderilen tüm iletilerin kalıcı olduğu ve 5 önceliğine sahip olması olabilir.

Bir uygulama, aşağıdaki örnekte gösterildiği gibi, MQTopic nesnesinin özelliklerini benzer bir şekilde ayarlayabilir:

```
MQTopic t1 = new MQTopic("Sport/Football/Results");
.
t1.setPersistence(WMQConstants.WMQ_PER_NON);
t1.setPriority(0);
```

Bu kod bir MQTopic nesnesi yaratır ve nesnenin iki özelliğini ayarlar. Bu özellikleri ayarlamaya ilişkin etki, hedefe gönderilen tüm iletilerin kalıcı olmayıp 0 önceliğine sahip olmalarıdır.

Bir hedefin özelliklerine ve özelliklerini ayarlamak için kullanılan yöntemlere ilişkin ek bilgi için IBM WebSphere MQ classes for JMS nesnelerinin özellikleribaşlıklı konuya bakın.

JMS uygulamasında bağlantı oluşturulması

Bir bağlantı oluşturmak için bir JMS uygulaması, Connection nesnesi yaratmak için bir ConnectionFactory nesnesini kullanır ve sonra bağlantıyı başlatır.

Bir uygulama, Connection nesnesi yaratmak için aşağıdaki örnekte gösterildiği gibi ConnectionFactory nesnesinin createConnection() yöntemini kullanır:

```
ConnectionFactory factory;
Connection connection;
.
.
.
connection = factory.createConnection();
```

Bir JMS bağlantısı yaratıldığında, IBM WebSphere MQ classes for JMS , bir bağlantı tanıtıcısı (Hconn) yaratır ve kuyruk yöneticisiyle bir etkileşim başlatır.

QueueConnectionFactory arabirimi ve TopicConnectionFactory arabirimi her biri ConnectionFactory arabiriminden createConnection() yöntemini devralır. Bu nedenle, aşağıdaki örnekte gösterildiği gibi, etki alanına özgü bir nesne yaratmak için createConnection() yöntemini kullanabilirsiniz:

```
QueueConnectionFactory qcf;
Connection connection;
.
.
.
connection = qcf.createConnection();
```

Bu kod parçası bir QueueConnection nesnesi yaratır. Bir uygulama artık bu nesne üzerinde bir etki alanı bağımsız işlemi gerçekleştirebilir ya da yalnızca noktadan noktaya etki alanı için geçerli olan bir işlemdir. Ancak, uygulama yalnızca yayınlama/abone olma etki alanı için geçerli olan bir işlemi gerçekleştirmeye çalışırsa, şu iletiyle bir IllegalStateKural Dışı Durumu kural dışı durumu yayınlandı:

```
JMSMQ1112: Operation for a domain specific object was not valid.
           Operation createProducer() is not valid for type com.ibm.mq.jms.MQTopic
```

Bunun nedeni, bağlantının bir etki alanından belirli bir bağlantı üreticisinden oluşturulduğundan kaynaklanır.

Not: Uygulama işlemi tanıtıcısının kuyruk yöneticisine geçirilecek varsayılan kullanıcı kimliği olarak kullanıldığını unutmayın. Uygulama istemci taşıma kipinde çalışıyorsa, bu işlem tanıtıcısı, ilgili yetkilerle sunucuda var olmalıdır. Farklı bir kimliğin kullanılmasını istiyorsanız, createConnection(kullanıcı adı, parola) yöntemini kullanın.

JMS belirtimi, stopped durumunda bir bağlantının yaratıldığını belirtir. Bağlantı başlatılıncaya kadar, bağlantıyla ilişkili bir ileti tüketicisi hiçbir ileti alamaz. Bir uygulama, bağlantı başlatmak için, aşağıdaki örnekte gösterildiği gibi, Connection nesnesinin start () yöntemini kullanır:

```
connection.start();
```

JMS uygulamasında oturum yaratılması

Bir oturum yaratmak için, JMS uygulaması Connection nesnesinin createSession() yöntemini kullanır.

createSession() yönteminde iki değiştirge vardır:

1. Oturumun hareket edip edilmediğini belirten bir değiştirge
2. Oturuma ilişkin alındı bildirimi modunu belirten bir parametre

Örneğin, aşağıdaki kod, hareket eden bir oturum yaratır ve AUTO_RELSE ' in bir alındı bildirimi kipine sahiptir:

```
Session session;
.
boolean transacted = false;
session = connection.createSession(transacted, Session.AUTO_ACKNOWLEDGE);
```

Bir JMS oturumu yaratıldığında, IBM WebSphere MQ classes for JMS , bir bağlantı tanıtıcısı (Hconn) yaratır ve kuyruk yöneticisiyle bir etkileşim başlatır.

Bir Oturum nesnesi ve ondan yaratılan herhangi bir MessageProducer ya da MessageConsumer nesnesi, çok iş parçacıklı bir uygulamanın farklı iş parçacıkları tarafından koşutzamanlı olarak kullanılamaz. Bu nesnelerin eşzamanlı olarak kullanılmamasını sağlamanın en basit yolu, her iş parçacığı için ayrı bir Oturum nesnesi yaratmamaktadır.

JMS uygulamalarındaki hareket edilen oturumlar

JMS uygulamaları, ilk olarak hareket eden bir oturum yaratarak yerel işlemleri çalıştırabilir. Bir uygulama bir işlemi kesinleştirebilir ya da geri alabilir.

JMS uygulamaları yerel işlemleri çalıştırabilir. Yerel hareket, değişikliklerin yalnızca, uygulamanın bağlı olduğu kuyruk yöneticisinin kaynaklarında yapılan değişiklikleri içeren bir işlemdir. To run local transactions, an application must first create a transacted session by calling the createSession() method of a Connection object, specifying as a parameter that the session is transacted. Daha sonra, oturum içinde gönderilen ve alınan tüm iletiler bir işlem sırasına göre gruplandırılır. Bir işlem, uygulama başlattığında ya da gönderdiği iletileri geri alırken, işlem başlatıldığından bu yana aldığı iletiler sona erer.

Bir işlem kesinleştirilmek için, bir uygulama Oturum nesnesinin commit () yöntemini çağırır. Bir işlem kesinleştirildiğinde, işlem içinde gönderilen tüm iletiler diğer uygulamalara teslim edilebilmesi için kullanılabilir duruma gelir ve işlem içinde alınan tüm iletiler, ileti alışverişi sunucusunun bunları yeniden uygulamaya teslim etme girişiminde bulunmayabilmesi için kabul edilir. Noktadan noktaya iletişim alanında, ileti alışverişi sunucusu alınan iletileri kuyruklarından da kaldırır.

Bir işlemi geri yüklemek için, bir uygulama Oturum nesnesinin rollback () yöntemini çağırır. Bir hareket geriye işlendiğinde, işlem içinde gönderilen tüm iletiler ileti alışverişi sunucusu tarafından atılır ve işlem içinde alınan tüm iletiler, teslim için yeniden kullanılabilir duruma gelir. Noktadan noktaya iletişim alanında, alınan iletiler kuyruklarına geri alınır ve diğer uygulamalar tarafından görülebilir duruma gelir.

Uygulama, hareket eden bir oturum yarattığında ya da commit () ya da rollback () yöntemini çağırdığında otomatik olarak yeni bir işlem başlar. Bu nedenle, hareket eden bir oturumun her zaman etkin bir hareketi vardır.

Bir uygulama hareket edilen bir oturumu kapattığında, örtük bir geri alma gerçekleşir. Bir uygulama bağlantıyı kapattığında, tüm bağlantının hareket ettiği oturumlar için örtük bir geri alma gerçekleşir.

Bir uygulama bağlantıyı kapatmadan sona ererse, tüm bağlantının hareket ettiği oturumlar için örtük bir geri alma da gerçekleşir.

Hareket, hareket eden bir oturumda tamamen içerilidir. Bir hareket oturumlara yayılamaz. Diğer bir deyişle, bir uygulamanın iki ya da daha fazla sayıda dönüştürücü oturumuna ileti gönderip alması ve sonra tüm bu işlemleri tek bir işlem olarak kesinleştirmesi ya da geriye işlemeleri mümkün değildir.

JMS oturumlarına ilişkin alındı bildirimi kipleri

İletilmeyen her oturumda, uygulamanın aldığı iletilerin nasıl kabul edildiğine karar veren bir alındı bildirimi kipi vardır. Üç alındı bildirimi kipi vardır ve alındı bildirimi kipinin seçimi, uygulamanın tasarımını etkiler.

Bir oturum hareket edilmezse, uygulamanın aldığı iletilerin kabul edilme şekli, oturumun alındı bildirimi kipiyle belirlenir. Üç alındı bildirimi kipleri aşağıdaki paragraflarda açıklanmaktadır:

OTO_BILDIR

Oturum, uygulama tarafından alınan her iletiyi otomatik olarak kabul eder.

İletiler uygulamaya zamanuyumlu olarak teslim ediliyorsa, bir Receive çağrısı her başarıyla tamamlandığında, oturum bir iletinin alındığını kabul eder. İletilerin zamanuyumsuz olarak teslim edilmesi durumunda, bir ileti dinleyicisinin onMessage() yöntemine her çağrı başarıyla tamamlanırsa, oturum bir iletinin alındığını kabul eder.

Uygulama başarıyla bir ileti alırsa, ancak hata onayının oluşmasını önlüyorsa, ileti teslim için kullanılabilir duruma gelir. Bu nedenle, uygulamanın yeniden teslim edilen bir iletiyi işleyebilmesi gerekir.

DUPS_OK_BILDIR

Bu oturum, uygulama tarafından alınan iletilerin seçim süresinde kabul ettiği kabul eder.

Bu alındı bildirimi kipinin kullanılması, oturumun yapması gereken iş miktarını azaltır, ancak ileti onayını önleyen bir hata, birden çok iletinin teslim edilmesi için kullanılabilir duruma gelmesinden kaynaklanabilir. Bu nedenle, uygulamanın yeniden teslim edilen iletileri işleyebilmesi gerekir.

Sınırlama: AUTO_RASK ve DUPS_OK_RECTID kiplerinde, JMS, ileti dinleyicisinde işlenemeyen bir kural dışı durum yayınlayarak bir uygulamayı desteklemez. Bu, ileti dinleyici döndüğünde, başarıyla işlenip işlenmediğine bakılmaksızın, ileti dinleyici döndüğünde iletilerin her zaman kabul ettiği anlamına gelir (hatalar onarılmaz değildir ve uygulamanın devam etmemesini engellememektedir). İleti onayının daha iyi bir şekilde denetlenmesini istiyorsanız, alındı bildirimi işlevlerinin uygulamaya tam denetimini veren CLIENT_RELIND ya da transated kiplerini kullanın.

CLIENT_ALIND

Uygulama, İleti sınıfının Acknowy yöntemini çağırarak aldığı iletileri kabul eder.

Uygulama, her iletinin alınmasını tek tek onaylayabilir ya da bir ileti kümesi alabilir ve Acknowy yöntemini yalnızca aldığı son ileti için çağırabilir. Yöntemin çağrıldığı son çağrıdan bu yana Acknowy yöntemi çağrıldığında alınan tüm iletiler onaylanır.

Bu onay kiplerinden herhangi biriyle birlikte, bir uygulama Oturum sınıfının Recon yöntemini çağırarak, bir oturumda iletilerin teslim edilmesini durdurabilir ve yeniden başlatabilir. Alınan ancak daha önce kabul edilmeyen iletiler yeniden teslim edilir. Ancak, bunlar daha önce teslim edildikleri sırayla teslim edilmeyebilir. Bu arada, daha yüksek öncelikli iletiler gelmiş olabilir ve özgün iletilerin bazılarının süresi dolmuş olabilir. Noktadan noktaya iletişim alanında, özgün iletilerin bazıları başka bir uygulama tarafından tüketilmiş olabilir.

Bir uygulama, iletinin JMSReverulamiş üstbilgi alanının içeriğini inceleyerek bir iletinin yeniden teslim edilip edilmediğini belirleyebilir. Uygulama bunu, Message sınıfının getJMSRedelivered() yöntemini çağırarak gerçekleştirir.

Bir JMS uygulamasında hedefler oluşturma

Bir Java Adlandırma ve Dizin Arabirimi (JNDI) ad alanından yönetilen nesneler olarak hedefleri almak yerine, bir JMS uygulaması yürütme sırasında dinamik dinamik hedefler yaratmak için bir oturum kullanabilir. An application can use a uniform resource identifier (URI) to identify a WebSphere MQ queue or a topic and, optionally, to specify one or more properties of a Queue or Topic object.

Kuyruk Nesneleri Yaratmak için Oturumun Kullanılması

Bir kuyruk nesnesi yaratmak için, bir uygulama bir Oturum nesnesinin createQueue() yöntemini kullanarak aşağıdaki örnekte gösterildiği gibi kullanabilir:

```
Session session;
.
Queue q1 = session.createQueue("Q1");
```

Bu kod, tüm özellikleri için varsayılan değerleri içeren bir Kuyruk nesnesi yaratır. Nesne, yerel kuyruk yöneticisine ait Q1 adlı bir WebSphere MQ kuyruğunu temsil eder. Bu kuyruk, yerel bir kuyruk, bir diğer ad kuyruğu ya da uzak kuyruk tanımlaması olabilir.

createQueue() yöntemi, parametre olarak bir kuyruk URI değerini de kabul eder. Kuyruk URI 'si, bir WebSphere MQ kuyruğunun adını ve isteğe bağlı olarak, kuyruğun sahibi olan kuyruk yöneticisinin adını ve Kuyruk nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki deyimde bir kuyruk URI örneği yer almaktadır:

```
Queue q2 = session.createQueue("queue://QM2/Q2?persistence=2&priority=5");
```

The Queue object created by this statement represents a WebSphere MQ queue called Q2 that is owned by a queue manager called QM2, and all messages sent to this destination are persistent and have a priority of 5. Bu şekilde tanımlanan kuyruk yöneticisi, yerel kuyruk yöneticisi ya da uzak kuyruk yöneticisi olabilir. If it is a remote queue manager, WebSphere MQ must be configured so that, when the application sends a message to this destination, Websphere MQ can route the message from the local queue manager to queue manager QM2. URI ' ler hakkında daha fazla bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844.

createQueue() yöntemindeki parametrenin sağlayıcıya özgü bilgiler içerdiğini unutmayın. Bu nedenle, bir Kuyruk nesnesi yaratmak için createQueue() yöntemini kullanarak, JNDI ad alanından yönetilen nesne olarak bir Kuyruk nesnesini almak yerine, uygulamanızı daha az taşınabilir hale getirebilirsiniz.

Bir uygulama, aşağıdaki örnekte gösterildiği gibi, bir Oturum nesnesinin createTemporaryQueue () yöntemini kullanarak bir TemporaryQueue nesnesi yaratabilir.

```
TemporaryQueue q3 = session.createTemporaryQueue();
```

Geçici bir kuyruk yaratmak için bir oturum kullanılsa da, geçici kuyruğun kapsamı, oturumu yaratmak için kullanılan bağlantıdır. Bağlantı oturumlarından herhangi biri, geçici kuyruk için ileti üreticileri ve ileti tüketicileri oluşturabilir. Geçici kuyruk, bağlantı sona erinceye ya da uygulama geçici kuyruğu belirtik olarak TemporaryQueue.delete () yöntemini kullanarak silinceye kadar kalır.

Bir uygulama geçici bir kuyruk yarattığında, JMS için WebSphere MQ sınıfları, uygulamanın bağlı olduğu kuyruk yöneticisinde dinamik bir kuyruk yaratır. Bağlantı üreticisinin TEMPMODEL özelliği, dinamik kuyruğu yaratmak için kullanılan model kuyruğunun adını ve bağlantı üreticisinin TEMPQPREFIX özelliğinin, dinamik kuyruğun adını oluşturmak için kullanılan öneki belirtmesini sağlar.

Konu nesneleri yaratmak için oturum kullanılması

Bir Konu nesnesi yaratmak için, bir uygulama bir Oturum nesnesinin createTopic() yöntemini kullanarak aşağıdaki örnekte gösterildiği gibi kullanabilir:

```
Session session;
.
Topic t1 = session.createTopic("Sport/Football/Results");
```

Bu kod, tüm özellikleri için varsayılan değerleri içeren bir Konu nesnesi yaratır. Nesne, Sport/Football/Results adlı bir konuyu gösterir.

createTopic() yöntemi bir konu URI 'sini parametre olarak da kabul eder. Konu URI 'si, bir konunun adını ve isteğe bağlı olarak, Konu nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Aşağıdaki kod, bir konu URI örneğini içerir:

```
String uri = "topic://Sport/Tennis/Results?persistence=1&priority=0";
Topic t2 = session.createTopic(uri);
```

Bu kod tarafından oluşturulan Konu nesnesi, Sport/Tennis/Results adlı bir konuyu temsil eder ve bu hedefe gönderilen tüm iletiler kalıcı değildir ve 0 önceliğine sahiptir. Konu URI ' leri hakkında daha fazla bilgi için bkz. “Birörnek kaynak tanıtıcıları (URI ' ler)” sayfa 844.

createTopic() yöntemindeki parametrenin sağlayıcıya özgü bilgiler içerdiğini unutmayın. Bu nedenle, bir Konu nesnesi yaratmak için createTopic() yöntemini kullanarak, bir JNDI ad alanından bir Topic nesnesini denetimli nesne olarak almak yerine, uygulamanızı daha az taşınabilir hale getirebilirsiniz.

Bir uygulama, aşağıdaki örnekte gösterildiği gibi, bir Oturum nesnesinin createTemporaryTopic () yöntemini kullanarak bir TemporaryTopic nesnesi oluşturabilir.

```
TemporaryTopic t3 = session.createTemporaryTopic();
```

Geçici bir konu yaratmak için bir oturum kullanılsa da, geçici bir konunun kapsamı, oturumu yaratmak için kullanılan bağlantıdır. Bağlantı oturumlarından herhangi biri geçici konu için ileti üreticileri ve ileti tüketicileri oluşturabilir. The temporary topic remains until the connection ends or the application explicitly deletes the temporary topic by using the TemporaryTopic.delete() method, whichever is the sooner.

Bir uygulama geçici bir konu yarattığında, JMS için WebSphere MQ sınıfları, TEP/*tempTopicPrefix*karakterleriyle başlayan bir adla bir konu yaratır; burada *tempTopicPrefix* , bağlantı üreticisinin TEMPTOPICFIX özelliğinin değeridir.

Birörnek kaynak tanıtıcıları (URI ' ler)

Kuyruk URI 'si, bir WebSphere MQ kuyruğunun adını ve isteğe bağlı olarak, kuyruğa sahip olan kuyruk yöneticisinin adını ve uygulama tarafından yaratılan kuyruk nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir. Konu URI 'si, bir konunun adını ve isteğe bağlı olarak, uygulama tarafından oluşturulan Konu nesnesinin bir ya da daha fazla özelliğini belirten bir dizgidir.

Kuyruk URI 'si şu biçimde olur:

```
queue://[qMgrName]/qName[?propertyName1=propertyValue1
                         &propertyName2=propertyValue2
                         &...]
```

Bir konu URI 'si şu biçimde olur:

```
topic://topicName[?propertyName1=propertyValue1
                  &propertyName2=propertyValue2
                  &...]
```

Bu biçimlerdeki değişkenler aşağıdaki anlamlara sahiptir:

qMgrAdı
URI ' nin tanımladığı kuyruğa sahip olan kuyruk yöneticisinin adı.

Kuyruk yöneticisi yerel kuyruk yöneticisi ya da uzak kuyruk yöneticisi olabilir. Uzak bir kuyruk yöneticisiyse, WebSphere MQ yapılandırılmış olmalıdır; böylece, bir uygulama kuyruğa ileti gönderdiğinde, Websphere MQ iletiyi yerel kuyruk yöneticisinden uzak kuyruk yöneticisine yönlendirebilir.

Ad belirtilmezse, yerel kuyruk yöneticisi varsayılan değer olarak kabul edilir.

qName
WebSphere MQ kuyruğunun adı.

Kuyruk, yerel bir kuyruk, bir diğer ad kuyruğu ya da uzak kuyruk tanımlaması olabilir.

Kuyruk adı yaratılmasına ilişkin kurallar için bkz. IBM WebSphere MQ nesnelerinin adlandırılmasına ilişkin kurallar .

topicName
Konunun adı.

Konu adlarının yaratılmasına ilişkin kurallar için bkz. IBM WebSphere MQ nesnelerinin adlandırılmasına ilişkin kurallar. +, #, * ve? joker karakterlerini kullanmaktan kaçının. konu adlarında. Bu karakterleri içeren konu adları, bunlara abone olduğunuzda beklenmeyen sonuçlara neden olabilir. Konu dizgilerinin kullanılmasıbaşlıklı konuya bakın.

***propertyName1*, *propertyName2*, ...**
Uygulama tarafından yaratılan Kuyruk ya da Konu nesnesinin özelliklerinin adları. Çizelge 123 sayfa 845 , bir URI ' de kullanılabilecek geçerli özellik adlarını listeler.

Hiçbir özellik belirtilmemişse, Kuyruk ya da Konu nesnesi tüm özellikleri için varsayılan değerlere sahiptir.

***propertyValue1*, *propertyValue2*, ...**
Uygulama tarafından yaratılan Kuyruk ya da Konu nesnesinin özelliklerinin değerleri. Çizelge 123 sayfa 845 , bir URI ' de kullanılabilecek geçerli özellik değerlerini listeler.

Köşeli ayraçlar ([]) isteğe bağlı bir bileşeni gösterir ve üç nokta (...), özellik ad-değer çiftleri listesinin bir ya da daha fazla ad-değer çifti içerebileceğini belirtir.

Çizelge 123 sayfa 845 içinde, geçerli özellik adları ve kuyruk ve konu URI ' lerinde kullanılabilecek geçerli değerler listelenir. WebSphere MQ JMS yönetim aracı, özelliklerin değerleri için simgesel sabitler kullansa da, URI ' ler simgesel sabitler içeremez.

Çizelge 123. Özellik adları ve kuyruk ve konu URI ' lerinde kullanım için geçerli değerler

Özellik adı	**Tanım**	**Geçerli değerler**
CCSID	JMS için WebSphere MQ sınıfları iletiyi hedefe iletirken bir iletinin gövdesindeki karakter verileri nasıl temsil edilir	• WebSphere MQtarafından desteklenen herhangi bir kodlanmış karakter takımı tanıtıcısı.
Kodlama	JMS için WebSphere MQ sınıfları iletiyi hedefe iletirken bir iletinin gövdesindeki sayısal verilerin nasıl temsil edilir	• Bir WebSphere MQ ileti tanımlayıcısında *Kodlama* alanı için geçerli herhangi bir değer.
Son kullanma tarihi	Hedefe gönderilen iletiler için yaşamanın zamanı	• -2-gönderme () çağrısında belirtildiği gibi ya da gönderme () çağrısında belirtilmediyse, ileti üreticisinin canlı olarak yaşaması için varsayılan süre. • 0-Hedefe gönderilen bir iletinin süresi hiçbir zaman sona ermez. • Milisaniye cinsinden canlanacak süreyi belirten pozitif bir tamsayı.

Çizelge 123. Özellik adları ve kuyruk ve konu URI ' lerinde kullanım için geçerli değerler (devamı var)		
Özellik adı	**Tanım**	**Geçerli değerler**
çok hedefli	Bir aracıya gerçek zamanlı bağlantı kullanılırken bir konuya ilişkin çoklu yayın ayarı	Aşağıdaki listede geçerli değerler yer almaktadır. Her bir değerle ilişkilendirilen, WebSphere MQ JMS yönetim aracında kullanılan MULTICAST özelliğinin karşılığı olan değerdir. MULTICAST özelliğinin ve geçerli değerlerinin bir açıklaması için IBM WebSphere MQ classes for JMS nesnelerinin özellikleribaşlıklı konuya bakın. • -1-ASCF • 0-DEVRE Dışı • 3-NOTR • 5-GÜVENILIR • 7-ETKIN
Kalıcılık	Hedefe gönderilen iletilerin sürekliliği	• -2-gönderme () çağrısında belirtildiği gibi ya da gönderme () çağrısında belirtilmediyse, ileti üreticisinin varsayılan kalıcılığı. • -1- WebSphere MQ kuyruğunda ya da konunun *DefPersistence* özniteliği tarafından belirtilir. • 1-Kalıcı olmayan. • 2-Kalıcı. • 3- WebSphere MQ JMS yönetim aracında kullanıldığı şekliyle PERSISTENCE özelliği için HIGH değerine eşittir. Bu değere ilişkin bir açıklama için bkz. “JMS kalıcı iletileri” sayfa 868.
öncelik	Hedefe gönderilen iletilerin önceliği	• -2-gönderme () çağrısında belirtildiği gibi ya da gönderme () çağrısında belirtilmediyse, ileti üreticisinin varsayılan önceliği. • -1- WebSphere MQ kuyruğunun ya da konunun *DefPriority* özniteliği tarafından belirtildiği gibi. • Hedefe gönderilen iletilerin önceliğini belirten 0-9 aralığında bir tamsayı.
targetClient	Hedefte gönderilen iletilerin bir MQRFH2 üstbilgisi içerip içermediğini	• 0-İletiler bir MQRFH2 üstbilgisi içerir. • 1-İletiler bir MQRFH2 üstbilgisi içermez.

Örneğin, aşağıdaki URI, yerel kuyruk yöneticisinin sahibi olduğu Q1 adlı bir WebSphere MQ kuyruğunu tanımlar. Bu URI kullanılarak yaratılan bir Kuyruk nesnesi, tüm özellikleri için varsayılan değerlere sahip olur.

```
queue:///Q1
```

The following URI identifies a WebSphere MQ queue called Q2 that is owned by a queue manager called QM2. Bu hedefe gönderilen tüm iletiler 6 önceliğine sahiptir. Bu URI kullanılarak yaratılan kuyruk nesnesinin diğer özellikleri varsayılan değerlerine sahip olmalıdır.

```
queue://QM2/Q2?priority=6
```

Aşağıdaki URI, Sport/Atlettics/Results adlı bir konuyu tanıtır. Bu hedefe gönderilen tüm iletiler kalıcı değildir ve 0 önceliğine sahiptir. Bu URI kullanılarak yaratılan Konu nesnesinin diğer özellikleri varsayılan değerlerine sahip olmalıdır.

```
topic://Sport/Athletics/Results?persistence=1&priority=0
```

JMS uygulamasına ileti gönderme

Bir JMS uygulamasının bir hedefe ileti gönderebilmesi için, önce hedef için bir MessageProducer nesnesi yaratmalıdır. Hedefe bir ileti göndermek için, uygulama bir ileti nesnesi yaratır ve MessageProducer nesnesinin gönderme () yöntemini çağırır.

Bir uygulama, iletileri göndermek için bir MessageProducer nesnesini kullanır. Bir uygulama, genellikle belirli bir hedef için bir MessageProducer nesnesi yaratır; bu nesne, ileti üreticisi kullanılarak gönderilen tüm iletilerin aynı hedefe gönderileceği şekilde bir kuyruk ya da konu olabilir. Bu nedenle, bir uygulamanın bir MessageProducer nesnesi yaratabilmesi için önce bir Kuyruk ya da Konu nesnesi yaratması gerekir. Bir Kuyruk ya da Konu nesnesinin nasıl yaratılacağı hakkında bilgi için aşağıdaki konulara bakın:

- “Bir JMS uygulamasında yönetilen nesneleri almak için JNDI kullanılıyor” sayfa 830
- “IBM JMS uzantılarını kullanma” sayfa 831
- “ WebSphere MQ JMS uzantılarını kullanma” sayfa 837
- “Bir JMS uygulamasında hedefler oluşturma” sayfa 843

Bir uygulama, MessageProducer nesnesi yaratmak için, aşağıdaki örnekte gösterildiği gibi, bir Oturum nesnesinin createProducer() yöntemini kullanır:

```
MessageProducer producer = session.createProducer(destination);
```

`destination` parametresi, uygulamanın önceden oluşturduğu bir Kuyruk ya da Konu nesnesidir.

Bir uygulamanın ileti gönderebilmesi için bir ileti nesnesi yaratması gerekir. Bir iletinin gövdesinde uygulama verileri bulunur ve JMS beş tip ileti gövdeyi tanımlar:

- Bayt
- Eşlem
- Nesne
- Akış
- Metin

Her ileti gövdesi tipinin, İleti arabiriminin bir alt arabirimi olan kendi JMS arabirimi ve bu tip bir gövde içeren bir ileti oluşturmak için Oturum arabirimindeki bir yöntem vardır. Örneğin, bir metin iletisi için arabirim TextMessageolarak adlandırılır ve bir uygulama, aşağıdaki deyimde gösterildiği gibi, bir oturum nesnesinin createTextMessage () yöntemini kullanarak bir metin iletisi kullanır:

```
TextMessage outMessage = session.createTextMessage(outString);
```

İletiler ve ileti gövdeleriyle ilgili daha fazla bilgi için bkz. “JMS iletileri” sayfa 775.

Bir ileti göndermek için, aşağıdaki örnekte gösterildiği gibi, bir uygulama bir MessageProducer nesnesinin gönderme () yöntemini kullanır:

```
producer.send(outMessage);
```

Bir uygulama gönderme () yöntemini her iki ileti sistemi etki alanındaki iletileri göndermek için kullanabilir. Hedefin doğası, hangi ileti sistemi etki alanının kullanılacağını belirler. However, TopicPublisher, the sub-interface of MessageProducer that is specific to the publish/subscribe domain, also has a publish() method, which can be used instead of the send() method. İki yöntem işlevsel olarak aynıdır.

Bir uygulama, belirlenmiş hedefi olmayan bir MessageProducer nesnesi yaratabilir. Bu durumda, gönderme () yöntemini çağırırken uygulamanın hedefi belirtmesi gerekir.

Bir uygulama bir işlem içinde ileti gönderirse, hareket kesinleştirilinceye kadar bu ileti hedefine teslim edilmez. Başka bir deyişle, bir uygulama ileti gönderemez ve aynı hareket içinde iletiye yanıt alır.

Bir hedef, bir uygulama bu iletiye ileti gönderdiğinde, JMS için WebSphere MQ sınıflarına iletilsin ve kuyruk yöneticisinin iletiyi güvenli bir şekilde alıp almadığını belirlemeksizin, denetimi yeniden uygulamaya geri döndürecek şekilde yapılandırılabilir. Bu, bazen *zamanuyumsuz koyma*olarak adlandırılır. Daha fazla bilgi için “JMS için IBM WebSphere MQ sınıflarında ileti zamanuyumsuz olarak ileti konması” sayfa 883 başlıklı konuya bakın.

JMS uygulamasında ileti alma

Bir uygulama iletileri almak için bir ileti tüketicisi kullanır. Dayanıklı bir konu abonesi, tüketici etkinlik dışı olduğunda gönderilenler de dahil olmak üzere, bir hedefe gönderilen tüm iletileri alan bir ileti tüketicisi 'dir. Bir uygulama, bir ileti seçiciyi kullanarak almak istediği iletileri seçebilir ve ileti dinleyicisi kullanılarak zamanuyumsuz iletiler alabilir.

Bir uygulama, iletileri almak için bir MessageConsumer nesnesini kullanır. Bir uygulama, belirli bir hedef için bir MessageConsumer nesnesi yaratır; bu nesne, ileti tüketicisi kullanılarak alınan tüm iletilerin aynı hedeften alınması için bir kuyruk ya da konu olabilir. Bu nedenle, bir uygulamanın MessageConsumer nesnesi yaratabilmesi için önce bir Kuyruk ya da Konu nesnesi yaratması gerekir. Bir Kuyruk ya da Konu nesnesinin nasıl yaratılacağı hakkında bilgi için aşağıdaki konulara bakın:

- “Bir JMS uygulamasında yönetilen nesneleri almak için JNDI kullanılıyor” sayfa 830
- “IBM JMS uzantılarını kullanma” sayfa 831
- “ WebSphere MQ JMS uzantılarını kullanma” sayfa 837
- “Bir JMS uygulamasında hedefler oluşturma” sayfa 843

Bir uygulama, MessageConsumer nesnesi yaratmak için, aşağıdaki örnekte gösterildiği gibi, bir Oturum nesnesinin createConsumer() yöntemini kullanır:

```
MessageConsumer consumer = session.createConsumer(destination);
```

`destination` parametresi, uygulamanın önceden oluşturduğu bir Kuyruk ya da Konu nesnesidir.

Daha sonra, uygulama, aşağıdaki örnekte gösterildiği gibi, hedeften bir ileti almak için MessageConsumer nesnesinin alma () yöntemini kullanır:

```
Message inMessage = consumer.receive(1000);
```

Alma () çağrısındaki parametre, hemen kullanılabilir bir ileti olmadığında, yöntemin uygun bir ileti için ne kadar süreyle bekleyeceğini belirtir. Bu parametreyi atlarsanız, uygun bir ileti gelene kadar arama blokları süresiz olarak ertelenmektedir. Uygulamanın bir ileti için beklemesini istemiyorsanız, bunun yerine receiveNoBekle () yöntemini kullanın.

Receive () yöntemi, belirli tipte bir ileti döndürür. Örneğin, bir uygulama bir metin iletisi aldığında, receive () çağrısının döndürdüğü nesne bir TextMessage nesnesi olur.

Ancak, bir receive () çağrısı tarafından döndürülen bildirilen nesne tipi bir ileti nesnesidir. Bu nedenle, yeni alınan bir iletinin gövdesinden verileri almak için, uygulamanın Message sınıfından daha belirli bir alt sınıfa (örneğin, TextMessage) dönüşümü gerekir. İletinin tipi bilinmiyorsa, uygulama tipini belirlemek için

instanceof işlecini kullanabilir. Bir uygulamanın, dönüştürmeden önce bir iletinin tipini belirlemesi her zaman iyi bir uygulamadır; böylece hatalar düzgün bir şekilde işlenebilir.

Aşağıdaki kod, instanceof işlecini kullanır ve bir metin iletisinin gövdesinden verilerin nasıl çıkarılacağını gösterir:

```
if (inMessage instanceof TextMessage) {
  String replyString = ((TextMessage) inMessage).getText();
  .
  .
  .
} else {
  // Print error message if Message was not a TextMessage.
  System.out.println("Reply message was not a TextMessage");
}
```

Bir uygulama bir işlem içinde ileti gönderirse, hareket kesinleştirilinceye kadar bu ileti hedefine teslim edilmez. Başka bir deyişle, bir uygulama ileti gönderemez ve aynı hareket içinde iletiye yanıt alır.

İleti tüketicisi, ileriye okuma için yapılandırılmış bir hedeften ileti alırsa, uygulama sona erdiğinde okuma öncesinde okuma arabelleğindeki kalıcı olmayan iletiler atılır.

Yayınlama/abone olma etki alanında, JMS iki tip ileti tüketicisi, kalıcı olmayan konu abonesi ve dayanıklı konu abonesi (aşağıdaki iki bölümde açıklanan) tanımlanmaktadır.

Dayanıklı olmayan konu aboneleri

Kalıcı olmayan bir konu abonesi yalnızca abone etkin durumdayken yayınlanan iletileri alır. Kalıcı olmayan abonelik, uygulama, dayanıklı olmayan bir konu abonesi oluşturduğunda ve uygulama aboneyi kapattığında ya da abonenin kapsamı dışlandığında sona erdiğinde başlar. JMS için WebSphere MQ sınıflarında bir uzantı olarak, kalıcı olmayan bir konu abonesi de alıkonan yayınları alır, ancak bir aracıya gerçek zamanlı bağlantı kullanıldığında bu yayınları almaz.

Kalıcı olmayan bir konu abonesi yaratmak için, uygulama etki alanı bağımsız createConsumer() yöntemini kullanabilir ve hedef olarak bir Konu nesnesi belirtebilirsiniz. Diğer bir seçenek olarak, bir uygulama etki alanını belirli bir createSubscriber() yöntemini kullanarak aşağıdaki örnekte gösterildiği gibi kullanabilir:

```
TopicSubscriber subscriber = session.createSubscriber(topic);
```

topic parametresi, uygulamanın önceden oluşturduğu bir Konu nesnesidir.

Dayanıklı konu aboneleri

Sınırlama: Bir uygulama, bir aracıya gerçek zamanlı bağlantı kullanırken, dayanıklı konu aboneleri oluşturamaz.

Dayanıklı bir konu abonesi, kalıcı bir aboneliğin ömrü boyunca yayınlanan tüm iletileri alır. Bu iletiler, abone etkin olmamakla birlikte, yayınlananlar arasında yer alır. JMS için WebSphere MQ sınıflarında bir uzantı olarak, kalıcı bir konu abonesi de alıkonan yayınları alır.

Bir uygulama, kalıcı bir konu abonesi oluşturmak için aşağıdaki örnekte gösterildiği gibi bir Oturum nesnesinin createDurableSubscriber () yöntemini kullanır:

```
TopicSubscriber subscriber = session.createDurableSubscriber(topic, "D_SUB_000001");
```

createDurableSubscriber () çağrısında ilk parametre, uygulamanın daha önce yarattığı bir Konu nesnesidir ve ikinci parametre, sürekli aboneliği tanımlamak için kullanılan bir addır.

Kalıcı bir konu abonesi yaratmak için kullanılan oturumun ilişkili bir istemci tanıtıcısı olmalıdır. Bir oturumla ilişkili istemci tanıtıcısı, oturumu yaratmak için kullanılan bağlantıya ilişkin istemci tanıtıcısıyla aynıdır. İstemci tanıtıcısı, ConnectionFactory nesnesinin CLIENTID özelliği ayarlanarak belirlenebilir. Diğer bir seçenek olarak, bir uygulama, Connection nesnesinin setClientID () yöntemini çağırarak istemci tanıtıcısını belirtebilir.

Kalıcı bir aboneliği tanımlamak için kullanılan ad, yalnızca istemci tanıtıcısı içinde benzersiz olmalıdır; dolayısıyla, istemci tanıtıcısı, kalıcı bir aboneliğin tam, benzersiz tanıtıcısının bir parçası olur. Daha önce yaratılmış bir kalıcı aboneliği kullanmaya devam etmek için, bir uygulama, sürekli abonelikle ilişkilendirilmiş aynı istemci tanıtıcısına sahip bir oturum kullanarak ve aynı abonelik adını kullanarak, dayanıklı bir konu abonesi oluşturmalıdır.

Kalıcı abonelik, bir uygulama, şu anda hiçbir kalıcı aboneliği olmayan bir istemci tanıtıcısını ve abonelik adını kullanarak dayanıklı bir konu abonesi oluşturduğunda başlar. Ancak, kalıcı bir abonelik, uygulama dayanıklı konu abonesi kapatıldığında sona ermez. Bir uygulamanın, kalıcı aboneliği sona erdirmek için, kalıcı abonelikle ilişkilendirilmiş istemci tanıtıcısına aynı olan bir Oturum nesnesinin unsubnest () yöntemini çağırması gerekir. unsubscreen () çağrısındaki parametre, aşağıdaki örnekte gösterildiği gibi abonelik adıdır:

```
session.unsubscribe("D_SUB_000001");
```

Dayanıklı bir aboneliğin kapsamı bir kuyruk yöneticidir. Bir kuyruk yöneticisinden kalıcı bir abonelik varsa ve başka bir kuyruk yöneticisine bağlı bir uygulama aynı istemci tanıtıcısı ve abonelik adına sahip kalıcı bir abonelik yarattıysa, iki kalıcı abonelik tamamen bağımsızdır.

İleti seçicileri

Bir uygulama, art arda gelen alma () çağrıları tarafından yalnızca belirli ölçütlere uyan iletilerin döndürülmesini belirtebilir. Bir MessageConsumer nesnesi oluştururken, uygulama hangi iletilerin alınacağını belirleyen bir SQL (Yapılandırılmış Sorgu Dili) ifadesi belirleyebilir. Bu SQL ifadesine *ileti seçici*adı verilir. İleti seçici, JMS ileti üstbilgisi alanlarının adlarını ve ileti özelliklerini içerebilir. Bir ileti seçicinin nasıl oluşturulacağı hakkında bilgi için bkz. “JMS ' de ileti seçicileri” sayfa 775.

Aşağıdaki örnekte, bir uygulamanın myPropadlı kullanıcı tanımlı bir özelliğe dayalı iletileri nasıl seçebileceği gösterilmektedir:

```
MessageConsumer consumer;
.
consumer = session.createConsumer(destination, "myProp = 'blue'");
```

JMS belirtimi, bir uygulamanın ileti tüketicisinin ileti seçicisini değiştirmesine izin vermiyor. Bir uygulama, ileti seçicisiyle bir ileti tüketicisi yarattıktan sonra, ileti seçici o tüketicinin ömrü boyunca kalır. Bir uygulama birden çok ileti seçiciyi gerektiriyorsa, uygulamanın her ileti seçici için bir ileti tüketicisi yaratması gerekir.

Bir uygulama Sürüm 7 kuyruk yöneticisine bağlı olduğunda, bağlantı üreticisinin MSGSELECTION özelliğine ilişkin bir etki göstermez. Performansı en iyi duruma getirmek için, tüm ileti seçimi kuyruk yöneticisi tarafından gerçekleştirilir.

Yerel yayınların engelleniyor

Bir uygulama, tüketicinin kendi bağlantısında yayınlanan yayınları yoksayan bir ileti tüketicisi yaratabilir. Uygulama, aşağıdaki örnekte gösterildiği gibi, bir createConsumer() çağrısında üçüncü parametreyi `true`olarak ayarlanarak yapar:

```
MessageConsumer consumer = session.createConsumer(topic, null, true);
```

Bir createDurableSubscriber () çağrısında, aşağıdaki örnekte gösterildiği gibi, uygulama dördüncü parametreyi `true`değerine ayarlayarak bunu yapar.

```
String selector = "company = 'IBM'";
TopicSubscriber subscriber = session.createDurableSubscriber(topic, "D_SUB_000001",
                                                              selector, true);
```

İletilerin zamanuyumsuz teslim edilmesi

Bir uygulama, ileti tüketicisi bir ileti dinleyicisini kaydettirerek, iletileri zamanuyumsuz olarak alabilir. İleti dinleyicisinin onMessageadlı bir yöntemi vardır. Bu yöntem, uygun bir ileti varsa ve amacı iletiyi işlemek olan, zamanuyumsuz olarak çağrılır. Aşağıdaki kod mekanizmayı gösterir:

```
import javax.jms.*;

public class MyClass implements MessageListener
{
  // The method that is called asynchronously when a suitable message is available
  public void onMessage(Message message)
  {
    System.out.println("Message is "+message);

    // The code to process the message
    .
    .
    .
  }
}
.
.
.
// Main program (possibly in another class)
.
// Creating the message listener
MyClass listener = new MyClass();

// Registering the message listener with a message consumer
consumer.setMessageListener(listener);

// The main program now continues with other processing
```

Bir uygulama, alma () çağrılarını zamanuyumlu olarak almak ya da ileti dinleyicilerini kullanarak zamanuyumsuz iletiler almak için bir oturumu kullanabilir, ancak her ikisi için de zamanuyumsuz bir oturum kullanabilir. Bir uygulamanın iletileri zamanuyumlu olarak ve zamanuyumsuz olarak alması gerekiyorsa, ayrı oturumlar yaratmalıdır.

Bir oturum, iletileri zamanuyumsuz olarak almak için ayarlandıktan sonra, o oturumda ya da o oturumdan yaratılan nesnelerle ilgili olarak aşağıdaki yöntemler çağrılamaz:

- MessageConsumer.alma ()
- MessageConsumer.alma (uzun)
- MessageConsumer.receiveNoBekle ()
- Session.acknowledge()
- MessageProducer.gönderme (Hedef, İleti)
- MessageProducer.gönderme (Hedef, İleti, int, tamsayı, uzun)
- MessageProducer.gönderme (İleti)
- MessageProducer.gönderme (Message, int, int, long)
- Session.commit()
- Session.createBrowser(Kuyruk)
- Session.createBrowser(Kuyruk, Dizgi)
- Session.createBytesMessage()
- Session.createConsumer(Hedef)
- Session.createConsumer(Hedef, Dizgi, boole)
- Session.createDurableSubscriber(Konu, Dizgi)
- Session.createDurableSubscriber(Konu, Dizgi, Dizgi, boole)
- Session.createMapMessage()
- Session.createMessage()
- Session.createObjectMessage()

- Session.createObjectMessage(Serializable)
- Session.createProducer(Hedef)
- Session.createQueue(Dizgi)
- Session.createStreamMessage()
- Session.createTemporaryQueue()
- Session.createTemporaryTopic()
- Session.createTextMessage()
- Session.createTextMessage(Dizgi)
- Session.createTopic()
- Session.getAcknowledgeMode()
- Session.getMessageListener()
- Session.getTransacted()
- Session.rollback()
- Session.unsubscribe(Dizgi)

Bu yöntemlerden biri çağrılırsa, iletiyi içeren bir JMS kural dışı durumu (JMSException):

```
JMSCC0033: Bir oturum zamanuyumsuz olarak kullanıldığında zamanuyumlu bir yöntem çağrısına izin
verilmez: 'yöntem adı'
```

atılır.

Zehirli iletiler alınıyor

Bir uygulama işlenemeyen bir iletiyi alabilir. İletinin işlenememesinin birkaç nedeni olabilir; örneğin, iletinin biçimi yanlış olabilir. Bu tür iletiler, etkili bir şekilde işlenmesini önlemek için, zehirli iletiler olarak tanımlanır ve özel işleme gerektirir.

Zehirli iletilerin nasıl işleneceği konusunda ayrıntılı bilgi için bkz. “Handling poison messages in IBM WebSphere MQ classes for JMS” sayfa 853.

V 7.5.0.8 Abonelik kullanıcı verilerinin alınması

IBM WebSphere MQ classes for JMS uygulamasının bir kuyruktan tüketmekte olduğu iletiler, yönetimsel olarak tanımlanmış bir kalıcı abonelik tarafından konulursa, uygulamanın, abonelikle ilişkili kullanıcı verileri bilgilerine erişmesi gerekir. Bu bilgi, iletiye özellik olarak eklenir.

Version 7.5.0, Fix Pack 8' tan, bir ileti MQPS klasörüyle RFH2 üstbilgisi içeren bir kuyruktan tüketildiğinde, Sud tuşuyla ilişkili değer (varsa), IBM WebSphere MQ classes for JMS uygulamasına döndürülen JMS İletisi nesnesine bir String özelliği olarak eklenir. To enable the retrieval of this property from the message, the constant JMS_IBM_SUBSCRIPTION_USER_DATA in the JmsConstants interface can be used with the method javax.jms.Message.getStringProperty(java.lang.String) to get the subscription user data.

In the following example, an administrative durable subscription is defined by using the MQSC command **DEFINE SUB**:

```
DEFINE SUB('MY.SUBCRIPTION') TOPICSTR('PUBLIC') DEST('MY.SUBSCRIPTION.Q')
USERDATA('Administrative durable subscription to put message to the queue MY.SUBSCRIPTION.Q')
```

PUBLIC konu dizgisine yayınlanan iletilerin kopyaları kuyruğa (MY.SUBSCRIPTION.Q) yerleştirilir. The user data that is associated with the durable subscription is then added as a property to the message, which is stored in the MQPS folder of the RFH2 header with the key Sud.

IBM WebSphere MQ classes for JMS uygulaması şunları arayabilir:

```
javax.jms.Message.getStringProperty(JmsConstants.JMS_IBM_SUBSCRIPTION_USER_DATA);
```

Daha sonra aşağıdaki Dizgi döndürülür:

```
Administrative durable subscription to put message to the queue MY.SUBSCRIPTION.Q
```

.

İlgili kavramlar

“MQRFH2 üstbilgisi ve JMS” sayfa 779

İlgili görevler

Yönetimle ilgili abonelik tanımlanması

İlgili başvurular

ALT

Arabirim JmsConstants

JMS uygulaması için WebSphere MQ sınıflarının kapatılması

JMS uygulaması için, durmadan önce belirli JMS nesnelerini belirtik olarak kapatabilmek için bir WebSphere MQ sınıfları için önem önemlidir. Finalizerler çağrılmayabilir, bu nedenle kaynaklara ücretsiz olarak güvenmeyin. Bir uygulamanın sıkıştırılmış izleme etkin ile sonlandırmasına izin verme.

Yalnızca bir uygulama, oturum düzeyinde ya da daha düşük düzeyde çok kısa ömürlü JMS nesneleri yarattıysa, yalnızca çöp toplama işlemi, JMS ve WebSphere MQ kaynakları için tüm WebSphere MQ sınıflarını zamanında yayınlayamaz. Bu nedenle, bir uygulamanın bir Bağlantı, Oturum, MessageConsumerya da MessageProducer nesnesini kapatması artık gerekli olmadığında çok önemlidir.

Bir uygulama Connection 'ı kapatmadan sona ererse, tüm bağlantının hareket ettiği oturumlar için örtük bir geri alma gerçekleşir. Uygulama tarafından yapılan değişikliklerin kesinleştirildiğinden emin olmak için, uygulamayı kapatmadan önce bağlantıyı açık bir şekilde kapatın.

Bir uygulamada JMS nesnelerini kapatmak için finalizerleri kullanmayın. Finaller çağrılmayacağından, kaynaklar serbest bırakılmayabilir. Bir Bağlantı kapatıldığında, bu bağlantı, içinden oluşturulan tüm Oturumları kapatır. Benzer şekilde, Oturum kapatıldığında bir oturumdan oluşturulan MessageConsumers ve MessageProducers kapatılır. Ancak, kaynakların zamanında serbest bırakılmasını sağlamak için, Oturum, MessageConsumersve MessageProducers ' u belirtik olarak kapatmayı düşünün.

İzleme sıkıştırması etkinleştirildiyse, System.Halt() kapanları ve olağandışı olağan dışı, denetimsiz JVM sonlandırmalarının bozuk bir izleme dosyasıyla sonuçlanabilir olması gerekir. Olanaklı olduğu durumlarda, gereksinim duyardığınız izleme bilgilerini topladığınızda izleme olanağını kapatın. Bir uygulamayı olağan dışı bir uca izliyorsanız, sıkıştırılmamış izleme çıkışını kullanın.

Handling poison messages in IBM WebSphere MQ classes for JMS

Bir zehir iletisi, alan bir MDB uygulaması tarafından işlenemeyen bir iletidir. Bir zehir iletiyle karşılaşılırsa, JMS MessageConsumer ve ConnectionConsumer nesneleri, bunu iki kuyruk özelliğine, BOQNAME ve BOTHRESH değerine göre yeniden istekte buluntabilirler.

Bazen, hatalı biçimlendirilmiş bir ileti kuyruğun üzerine gelir. Bu bağlamda kötü biçimlendirilmiş, alma uygulamasının iletiyi doğru işleyemediği anlamına gelir. Böyle bir ileti, alma uygulamasının başarısız olmasına ve bu hatalı biçimlendirilmiş iletiyi geri almasına neden olabilir. Daha sonra, ileti giriş kuyruğuna sürekli olarak teslim edilebilir ve uygulama tarafından sürekli olarak yedeklenebilir. Bu iletiler *zehirli iletiler*olarak bilinir. JMS MessageConsumer nesnesi, zehirli iletileri saptar ve diğer bir hedefe yönlendirir.

IBM WebSphere MQ kuyruk yöneticisi, her iletinin geriletilmesinin kaç kez kaydedildiğini kaydeder. Bu sayı yapılandırılabilir bir eşik değerine ulaştığında, ileti tüketicisi iletiyi adlandırılmış bir geri alma kuyruğuna sunan bir kuyruğa iletir. Bu istekte bulunanların herhangi bir nedenle başarısız olması durumunda, ileti giriş kuyruğundan kaldırılır ve istek-mektup kuyruğuna ya da kuyruktan atılır ya da atılır. Daha ayrıntılı bilgi için bkz. “ASF ' de kuyruktan iletilerin kaldırılması” sayfa 891 .

There is a difference between the way in which poison messages are requeued by MessageConsumers and ConnectionConsumers. ConnectionConsumers , ileti teslimini etkilemeden zehirli iletileri yeniden istekte bulunmayı başarmış durumda. İstek süreci, gerçek ileti teslimi ile ilişkili herhangi bir iş biriminin dışında, uygulama kodu ile gerçekleşir. Bu, ConnectionConsumer işleminin çok iş parçacıklı doğası nedeniyle mümkündür.

Ancak,MessageConsumers, Oturum düzeyinin altında tek iş parçacıklarıdır ve her türlü zehir iletisi, yürürlükteki iş birimi içinde gerçekleşir. Bu, uygulamanın çalışmasını etkilemez; ancak, zehir iletileri bir transated ya da Client_REN; Oturum altında yeniden kuyruğa alındığında, yürürlükteki iş birimi uygulama kodu tarafından kesinleştirilinceye ya da uygunsa, uygulama kapsayıcı kodunu gerçekleştirinceye kadar, istekte bulunanın kendisi işlenmez.

JMS ConnectionConsumer nesneleri, zehirli iletileri aynı şekilde ve aynı kuyruk özelliklerini kullanarak işler. Birden çok bağlantı uygulaması aynı kuyruğu izliyorsa, zehir iletisinin, istek gerçekleşmeden önce bir uygulamaya eşik değerinden daha fazla zaman sağlanabileceği olabilir. Bu davranış, tek tek bağlantı tüketicilerinin kuyrukları izlemesine ve zehirli iletilerin istekte bulunmaya yol göstermesine neden olur.

Eşik değeri ve arka çıkış kuyruğunun adı bir IBM WebSphere MQ kuyruğuna ilişkin özniteliklerdir. Özniteliklerin adları şunlardır: BackoutThreshold ve BackoutRequeueQName. Uyguladıkları kuyruk şu şekildir:

- Noktadan noktaya ileti sistemi için bu, temeldeki yerel kuyruğdur. Bu, ileti tüketicileri ve bağlantı okuyucuları kuyruk diğer adlarını kullandığında önemlidir.
- IBM WebSphere MQ ileti alışverişi sağlayıcısı olağan kipinde yayınlama/abone olma ileti sistemi için, Konu tarafından yönetilen kuyruk, model kuyruğundan yaratılır.
- IBM WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipinde yayınlama/abone olma ileti alışverişi için, bu, TopicConnectionFactory nesnesinde tanımlanmış CCSID ' dir ya da Konu nesnesinde tanımlanan CCDSUB kuyruğu.

IBM WebSphere MQ classes for JMS , kuyruğun BackoutThreshold ve BackoutRequeueQName 'i sorgular. Bu nedenle, uygulamayı çalıştıran kullanıcıya kuyruk için sorgu erişimi vermelisiniz.

V 7.5.0.9 Hedef kuyruk bir küme kuyruğalıysa, gereken yetkiler, kullanılmakta olan IBM WebSphere MQ classes for JMS sürümünün sürümüne bağlıdır:

- When using the IBM WebSphere MQ classes for JMS for Version 7.5.0, Fix Pack 9 plus an interim fix for APAR IT26482, inquire access is required.
- Diğer tüm sürümler için, sorgulama izni verin, göz atın ve erişim alın.

BackoutThreshold ve BackoutRequeueQName özniteliklerini ayarlamak için aşağıdaki MQSC komutunu verin:

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

BackoutThreshold özniteliği sıfırdan farklı bir değere ayarlıysa, beklenmeyen bir davranış oluşmasını önlemek için BackoutRequeueQName özniteliğini geçerli bir kuyruk adına ayarlayın.

Yayınlama/abone olma ileti sistemi için, sisteminiz her abonelik için bir dinamik kuyruk oluşturuyorsa, bu öznitelik değerleri IBM WebSphere MQ classes for JMS model kuyruğundan, SYSTEM.JMS.MODEL.QUEUE. Bu ayarları değiştirmek için şunu kullanın:

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

Geriletme eşiği değeri sıfırsa, ileti işleme işlevi devre dışı bırakılır ve veri girişi kuyruğunda zehirli iletiler kalır. Tersi durumda, geriletme sayısı eşik değerine ulaştığında, ileti, adı belirtilen geriletme kuyruğuna gönderilir. Geriletme sayısı eşik değerine ulaşırsa, ancak ileti geriletme kuyruğuna giremezse, ileti ölü mektup kuyruğuna gönderilir ya da atılır. Bu durum, geriletme kuyruğu tanımlanmadıysa ya da MessageConsumer nesnesi iletiyi çıkış kuyruğuna gönderemezse oluşur. Ek ayrıntılar için “ASF ' de kuyruktan iletilerin kaldırılması” sayfa 891 başlıklı konuya bakın.

Bir ileti, geriletme yeniden kuyruğa alma kuyruğuna istekte bulunduğunda, ileti değiştirgesindeki (MQMD) İleti Tanımlayıcısındaki (MQMD) alan değerlerinden bazıları değişir. MQMD ' nin biçimiyle ilgili ayrıntılar için MQMD-Message Descriptor başlıklı konuya bakın.

İzleyen MQMD alanları, ileti geriletme kuyruğuna girdiğinizde değeri değiştirir.

- PutDate , geriletme yeniden kuyruğa alma kuyruğunda olduğu tarihe güncellenir.
- PutTime , geriletme yeniden kuyruğa alma kuyruğuna gireceği zaman olarak güncellenir.

- Geri alma sayısı sıfıra sıfırlandı.
- İleti süre bitimi, özgün iletinin JMS uygulaması tarafından alındığı sırada kalan süre bitimini yansıtacak şekilde güncellenir.

İleti geriletme kuyruğuna girdiğinizde, aşağıdaki alanlardaki değerler aynı kalır:

- StructId
- S\u00fcr\u00fcm
- Rapor
- MessageType
- Geribildirim
- Kodlama
- CodedCharSetId
- MsgId
- CorrelId
- ReplyToQ
- ReplyToQMgr
- Biçim
- Kalıcılık
- Öncelik

IBM WebSphere MQ classes for JMSiçinde kural dışı durumlar

Bir IBM WebSphere MQ classes for JMS uygulaması, bir JMS API çağrıları tarafından yayınlanan ya da bir kural dışı durum işleyicisine teslim edilen kural dışı durumları işleyebilmelidir.

IBM WebSphere MQ classes for JMS , kural dışı durumlar yayınlayarak çalıştırma zamanı sorunlarını bildirir. JMSException, JMS yöntemleri tarafından yayınlanan kural dışı durumlar için kök sınıftır ve JMSException kural dışı durumlarını yakalamak, JMS ile ilgili tüm özel durumları işlemek için soysal bir yol sağlar.

Her JMSException kural dışı durumu aşağıdaki bilgileri sarsalıyor:

- Sağlayıcıya özgü bir kural dışı durum iletisi; bir uygulama, Throwable.getMessage() yöntemini çağırarak bu kural dışı durum iletisini alır.
- Sağlayıcıya özgü bir hata kodu; bir uygulama, JMSException.getErrorCode() yöntemini çağırarak bu kodu alır.
- Bağlantılı bir kural dışı durum. Bir JMS API çağrısı tarafından yayınlanan bir kural dışı durum, genellikle bu kural dışı duruma bağlı başka bir kural dışı durum tarafından bildirilen daha düşük düzeyli bir sorunun sonuçtır. Bir uygulama, JMSException.getLinkedException() öğesini ya da Throwable.getCause() yöntemini çağırarak, bağlantılı bir kural dışı durumu alır.

IBM WebSphere MQ classes for JMS tarafından verilen çoğu kural dışı durum, JMSException alt sınıflarının örnekleridir. Bu alt sınıflar, aşağıdaki ek bilgileri sağlayan com.ibm.msg.client.jms.JmsExceptionDetail arabirimini uygular.

- Bir uygulamanın JmsExceptionDetail.getExplanation() yöntemini çağırarak ilgili olduğu kural dışı durum iletisine ilişkin açıklama.
- Bir uygulamanın JmsExceptionDetail.getUserAction() yöntemini çağırarak ilgili olduğu kural dışı duruma yanıt veren bir kullanıcı yanıtı.
- Kural dışı durum iletisinde ileti eklerinin anahtarları. Bir uygulama, JmsExceptionDetail.getKeys() yöntemini çağırarak, tüm tuşlar için bir yineleyici alır.
- İleti, kural dışı durum iletisine eklenir. Örneğin, bir ileti ekleme işlemi, kural dışı duruma neden olan kuyruğun adı olabilir ve bir uygulamanın bu ada erişebilmesinin yararlı olabileceği bir uygulama olabilir. Bir uygulama, JmsExceptionDetail.getValue() yöntemini çağırarak, belirtilen bir anahtara karşılık gelen ileti eklenmesini alır.

Herhangi bir ayrıntı yoksa, JmsExceptionAyrıntı arabirimindeki tüm yöntemler boş değer döndürebilir.

Örneğin, bir uygulama var olmayan bir IBM WebSphere MQ kuyruğu için ileti üreticisi yaratmayı denerse, aşağıdaki bilgilerle bir kural dışı durum oluşur:

```
Message : JMSWMQ2008: Failed to open MQ queue 'Q_test'.
Class : class com.ibm.msg.client.jms.DetailedInvalidDestinationException
Error Code : JMSWMQ2008
Explanation : JMS attempted to perform an MQOPEN, but WebSphere MQ reported an
              error.
User Action : Use the linked exception to determine the cause of this error. Check
              that the specified queue and queue manager are defined correctly.
```

Kural dışı durum yayınlandı, com.ibm.msg.client.jms.DetailedInvalidDestinationException, javax.jms.InvalidDestinationException sınıfının alt sınıfıdır ve com.ibm.msg.client.jms.JmsExceptionDetail arabirimini gerçekleştirir.

Bağlantılı özel durumlar

Bağlantılı kural dışı durum, bir yürütme ortamı sorunuyla ilgili ek bilgi sağlar. Bu nedenle, yayınlanan her JMSException kural dışı durumu için, bir uygulamanın bağlantılı kural dışı durumu denetmesi gerekir. Bağlantılı kural dışı durumun kendisi başka bir bağlantılı kural dışı duruma sahip olabilir ve bu nedenle bağlantılı kural dışı durumlar, özgün temel soruna geri giden bir zincir oluşturur. java.lang.Throwable sınıfının zincirleme kural dışı durum mekanizması kullanılarak bağlantılı bir kural dışı durum uygulandı ve bir uygulama, Throwable.getCause() yöntemini çağırarak bağlantılı bir kural dışı durumu alır. Bir JMSException kural dışı durumu için, getLinkedException () yöntemi aslında Throwable.getCause() yönteminde yetki aktarır.

Örneğin, bir uygulama kuyruk yöneticisine bağlanırken yanlış bir kapı numarası belirtiyorsa, kural dışı durumlar aşağıdaki zinciri oluşturur:

```
com.ibm.msg.client.jms.DetailIllegalStateException
   |
   +--->com.ibm.mq.MQException
            |
            +--->com.ibm.mq.jmqi.JmqiException
                    |
                    +--->java.net.ConnectionException
```

Genellikle, bir zincirdeki her kural dışı durum, koddaki farklı bir katmandan atılır. Örneğin, yukarıdaki zincirdeki kural dışı durumlar aşağıdaki katmanlar tarafından atılır:

- JMSException 'ın alt sınıfının bir eşgörünümü olan ilk kural dışı durum, IBM WebSphere MQ classes for JMSiçindeki ortak katman tarafından atılır.
- Sonraki kural dışı durum, IBM WebSphere MQ ileti sistemi sağlayıcısı tarafından com.ibm.mq.MQExceptionörneği tarafından verilir.
- Bir sonraki kural dışı durum (com.ibm.mq.jmqi.JmqiException), ortak Java arabirimi tarafından MQI ' ye atılır.
- Son kural dışı durum, Java sınıf kitaplığı tarafından bir java.net.ConnectionExceptioneşgörünümü yayınlanıyor.

For more information about the layered architecture of IBM WebSphere MQ classes for JMS, see “JMS mimarisine ilişkinIBM WebSphere MQ sınıfları” sayfa 766.

Aşağıdaki kodla benzer kod kullanılarak, bir uygulama tüm uygun bilgileri ayıklamak için bu zincirden geçerek yinelenebilir:

```
import com.ibm.msg.client.jms.JmsExceptionDetail;
import com.ibm.mq.MQException;
import com.ibm.mq.jmqi.JmqiException;
import javax.jms.JMSException;
.
.
.
catch (JMSException je) {
    System.err.println("Caught JMSException");
```

```
    // Check for linked exceptions in JMSException
    Throwable t = je;
    while (t != null) {
        // Write out the message that is applicable to all exceptions
        System.err.println("Exception Msg: " + t.getMessage());
        // Write out the exception stack trace
        t.printStackTrace(System.err);

        // Add on specific information depending on the type of exception
        if (t instanceof JMSException) {
            JMSException je1 = (JMSException) t;
            System.err.println("JMS Error code: " + je1.getErrorCode());

            if (t instanceof JmsExceptionDetail){
                JmsExceptionDetail jed = (JmsExceptionDetail)je1;
                System.err.println("JMS Explanation: " + jed.getExplanation());
                System.err.println("JMS Explanation: " + jed.getUserAction());
            }
        } else if (t instanceof MQException) {
            MQException mqe = (MQException) t;
            System.err.println("WMQ Completion code: " + mqe.getCompCode());
            System.err.println("WMQ Reason code: " + mqe.getReason());
        } else if (t instanceof JmqiException){
            JmqiException jmqie = (JmqiException)t;
            System.err.println("WMQ Log Message: " + jmqie.getWmqLogMessage());
            System.err.println("WMQ Explanation: " + jmqie.getWmqMsgExplanation());
            System.err.println("WMQ Msg Summary: " + jmqie.getWmqMsgSummary());
            System.err.println("WMQ Msg User Response: "
                               + jmqie.getWmqMsgUserResponse());
            System.err.println("WMQ Msg Severity: " + jmqie.getWmqMsgSeverity());
        }

        // Get the next cause
        t = t.getCause();
    }
}
```

Bir uygulamanın bir zincirdeki her bir kural dışı durumun tipini her zaman denetleyeceğini unutmayın; kural dışı durum tipi farklı tipteki bilgileri sarmalayan farklı tiplerdeki kural dışı durumlar ve kural dışı durumlar olabilir.

IBM WebSphere MQ ile ilgili bir sorunla ilgili özel bilgilerin alınması

Instances of com.ibm.mq.MQException and com.ibm.mq.jmqi.JmqiException encapsulate IBM WebSphere MQ specific information about a problem.

Bir MQException kural dışı durumu, aşağıdaki bilgileri sarmalıyor:

- Bir uygulama, getCompCode () yöntemini çağırarak bir uygulama tarafından elde edilen bir kod
- Bir uygulama, bir uygulamanın getReason() yöntemini çağırarak edindiği bir neden kodu

Bir JmqiException kural dışı durumu, bir tamamlanma kodunu ve bir neden kodunu da sarsalıyor. Ancak ek olarak, bir JmqiException kural dışı durumu, bir AMQ*nnnn* ya da CSQ*nnnn* iletisinde, kural dışı durumla ilişkiliyse, bilgileri sarsalıyor. Bir uygulama, kural dışı durumun uygun yöntemlerini çağırarak, bu iletinin çeşitli bileşenlerini (önem derecesi, açıklama ve kullanıcı yanıtı gibi) alabilir.

Bu kısımda sözü edilen yöntemlerin kullanılmasına ilişkin örnekler için, “Bağlantılı özel durumlar” sayfa 856içindeki örnek kodlara bakın.

Önceki IBM WebSphere MQ classes for JMSsürümlerinden yükseltme

Önceki IBM WebSphere MQ classes for JMSsürümlerine göre, çoğu hata kodu ve kural dışı durum iletileri Sürüm 7 'de değişmektedir. Bu değişikliklerin nedeni, IBM WebSphere MQ classes for JMS ' in artık katmanlı bir mimariye sahip olması ve koddaki farklı katmanlardan kural dışı durumları ortaya atılmasına neden olur.

For example, if an application tries to connect to a queue manager that does not exist, a previous version of IBM WebSphere MQ classes for JMS threw a JMSException exception with the following information:

```
MQJMS2005: Failed to create MQQueueManager for 'localhost:QM_test'.
```

Bu kural dışı durum, aşağıdaki bilgilerle bağlantılı bir MQException kural dışı durumu içeriyordu:

```
MQJE001: Completion Code 2, Reason 2058
```

Aynı koşullarda karşılaştırmaya göre, IBM WebSphere MQ classes for JMS Sürüm 7, aşağıdaki bilgilerle JMSException kural dışı durumunu yayınlıyor:

```
Message : JMSWMQ0018: Failed to connect to queue manager 'QM_test' with
          connection mode 'Client' and host name 'localhost'.
Class : class com.ibm.msg.client.jms.DetailedJMSException
Error Code : JMSWMQ0018
Explanation : null
User Action : Check the queue manager is started and if running in client mode,
              check there is a listener running. Please see the linked exception
              for more information.
```

Bu kural dışı durum, aşağıdaki bilgilerle bağlantılı bir MQException kural dışı durumu içeriyor:

```
Message : JMSCMQ0001: WebSphere MQ call failed with compcode '2' ('MQCC_FAILED')
          reason '2058' ('MQRC_Q_MGR_NAME_ERROR').
Class : class com.ibm.mq.MQException
Completion Code : 2
Reason Code : 2058
```

Uygulamanız, Throwable.getMessage() yönteminin döndürdüğü kural dışı durum iletilerini ya da JMSException.getErrorCode() yöntemi tarafından döndürülen hata kodlarını sınarsa ve Sürüm 7 'den önce bir yayından yükseliyorsanız, uygulamanızın muhtemelen IBM WebSphere MQ classes for JMSSürüm 7 'yi kullanmak için değiştirilmesi gerekir.

Kural dışı durum

Bir uygulama, bir kural dışı durum dinleyicisini bir Bağlantı nesnesiyle kaydettirebilir. Subsequently, if a problem occurs that makes the connection unusable, IBM WebSphere MQ classes for JMS delivers an exception to the exception listener by calling its onException() method. Uygulama, daha sonra bağlantıyı yeniden oluşturma olanağına sahiptir.

V 7.5.0.8 APAR IT14820, included from IBM WebSphere MQ Version 7.5.0, Düzeltme Paketi 8, fixed a defect where an application's JMS ExceptionListener would not be invoked for non-connection broken exceptions (for example MQRC_GET_INHIBITED) even though the ASYNC_EXCEPTIONS property on the JMS Connection Factory used by the application, was set to ASYNC_EXCEPTIONS_ALL. Bu, Version 7.5.0, Fix Pack 8' dan önceki varsayılan değerdir.

V 7.5.0.8 Bir JMS MessageListener ve bir JMS ExceptionListeneryapılandıran ve IBM WebSphere MQ classes for JMS ' in JMS belirtimiyle tutarlı olduğundan emin olmak için, yürürlükteki JMS uygulamalarının davranışını korumak için, IBM WebSphere MQ classes for JMSiçin ASYNC_EXCEPTIONS JMS ConnectionFactory özelliğinin varsayılan değeri ASYNC_EXCEPTIONS_CONNECTIONBROKEN olarak değiştirildi. Sonuç olarak, varsayılan olarak, bir uygulamanın JMS ExceptionListener' a yalnızca bozuk bağlantı hata kodlarına karşılık gelen özel durumlar verilir.

V 7.5.0.8 From Version 7.5.0, Fix Pack 8, the IBM WebSphere MQ classes for JMS have also been updated such that JMSExceptions relating to non-connection broken errors, which occur during message delivery to asynchronous message consumers, are still delivered to a registered ExceptionListener when the JMS ConnectionFactory used by the application has the ASYNC_EXCEPTIONS property set to the value ASYNC_EXCEPTIONS_ALL.

V 7.5.0.8 Version 7.5.0, Fix Pack 8 için kural dışı durum dinleyicilerine ve değişikliklerin neden daha önceki yayınlardan yapılmış olduğuna ilişkin daha fazla bilgi için bkz. JMS: Exception listener changes in Version 7.5.

Diğer herhangi bir sorun için, yürürlükteki JMS API çağrısı tarafından JMSException kural dışı durumu yayınlandı.

Bir uygulama, bir Connection nesnesiyle bir kural dışı durum dinleyicisini kaydetmezse, kural dışı durum dinleyicisine teslim edilmiş olan kural dışı durumlar IBM WebSphere MQ classes for JMS günlüğüne yazılır.

İlgili başvurular
ASYNCEXCEPTION

JMS için WebSphere MQ sınıflarında günlüğe kaydetme hataları
Kullanıcı tarafından düzeltici işlem gerektirebilecek çalıştırma zamanı sorunları hakkında bilgi, JMS günlüğü için WebSphere MQ sınıflarına yazılır.

Örneğin, bir uygulama bir bağlantı üreticisinin bir özelliğini ayarlamaya çalışırsa, ancak özelliğin adı tanınmazsa, JMS için WebSphere MQ sınıfları, günlüğün soruna ilişkin bilgileri yazar.

Varsayılan değer olarak, günlüğü içeren dosyanın adı mqjms.log olarak adlandırılır ve yürürlükteki çalışma dizinidir. Ancak, JMS yapılandırma dosyasına ilişkin WebSphere MQ sınıflarında com.ibm.msg.client.commonservices.log.outputName özelliğini ayarlayarak, günlük dosyasının adını ve konumunu değiştirebilirsiniz. JMS yapılandırma dosyasına ilişkin WebSphere MQ sınıflarıyla ilgili bilgi için bkz. “IBM WebSphere MQ classes for JMS yapılandırma dosyası” sayfa 698ve com.ibm.msg.client.commonservices.log.outputName özelliğine ilişkin geçerli değerler hakkında daha fazla ayrıntı için bkz. “Günlüğe kaydetme ve IBM WebSphere MQ classes for JMS” sayfa 764.

JMS için WebSphere MQ sınıflarında ilk hata destek teknolojisi (FFST)
JMS için WebSphere MQ sınıflarında ciddi bir iç hata oluşursa, ilk hata destek teknolojisi (FFST) bilgileri oluşturulur.

FFST bilgileri, dosyaya JMSC*nnnn*adı verilir.FDC, burada *nnnn* dört basamaklı bir sayıdır. Bu dosya, izleme çıkışının yazıldığı dizinin altdizini olan FFDC (FFDC) adlı bir dizinde yer alıyor. Varsayılan olarak, izleme çıkışı yürürlükteki çalışma dizinine yazılır, ancak JMS yapılandırma dosyasına ilişkin WebSphere MQ sınıflarında **com.ibm.msg.client.commonservices.trace.outputName** özelliğini ayarlayarak izleme çıkışını farklı bir dizine yeniden yönlendirebilirsiniz. JMS yapılandırma dosyası için WebSphere MQ sınıflarıyla ilgili bilgi için bkz. “IBM WebSphere MQ classes for JMS yapılandırma dosyası” sayfa 698.

If tracing is enabled when FFT information is generated, the FFT information is also be written to the trace file. JMS programlarının izlenmesine ilişkin ek bilgi için IBM WebSphere MQ classes for JMS uygulamalarını izlemebaşlıklı konuya bakın.

FFDC kütüklerinin üretimini gizlemek için, **com.ibm.msg.client.commonservices.ffst.suppress**özelliğini aşağıdaki gibi ayarlayın:

0
Tüm FFDC kütüklerinin çıkışı (varsayılan).

-1
Çıkış yalnızca belirli bir tipteki ilk FFDC kütüklerinden birini içerir.

tamsayı
Bu sayıdan çok sayıda olanlar dışındaki tüm FFDC dosyalarını engelle.

JMS uygulaması için bir WebSphere MQ sınıflarından WebSphere MQ özelliklerine erişilmesi

WebSphere MQ classes for JMS provides facilities to exploit a number of features of WebSphere MQ.

Uyarı: Bu özellikler JMS belirtiminin dışındadır ya da bazı durumlarda JMS belirtimini ihlal eder. Bunları kullanırsanız, uygulamanızın diğer JMS sağlayıcılarıyla uyumlu olma olasılığı düşüktür. JMS belirtimine uygun olmayan özellikler, bir Uyarı bildirimi ile etiketlenir.

JMS uygulaması için bir WebSphere MQ sınıflarından ileti tanımlayıcısının okunması ve yazılması
Bir Hedef ve İleti üzerinde özellikleri ayarlayarak, ileti tanımlayıcısına (MQMD) erişme yeteneğini kontrol edin.

Bazı WebSphere MQ uygulamaları, kendilerine gönderilen iletilerin MQMD ' de ayarlanması için belirli değerler gerektirir. WebSphere MQ class for JMS, JMS uygulamalarının MQMD alanlarını ayarlamasına olanak sağlayan ileti özniteliklerini sağlar ve JMS uygulamalarının "sürücü" WebSphere MQ uygulamalarıyla etkinleştirilmesini sağlar.

MQMD özelliklerinin herhangi bir etkisine sahip olması için, WMQ_MQMD_WRITE_ENABLED hedef nesne özelliğini true değerine ayarlamalısınız. Daha sonra, MQMD alanlarına değer atamak için, iletinin özellik ayarı yöntemlerini (örneğin, setStringÖzelliği) kullanabilirsiniz. StrucId ve Sürüm dışında tüm MQMD alanları gösterilir; BackoutCount okunabilir ancak yazılamaz.

Bu örnek, MQMD.UserIdentifier , "JoeBloggs" olarak ayarlanır.

```
// Create a ConnectionFactory, connection, session, producer, message
// ...

// Create a destination
// ...

// Enable MQMD write
dest.setBooleanProperty(WMQConstants.WMQ_MQMD_WRITE_ENABLED, true);

// Optionally, set a message context if applicable for this MD field
dest.setIntProperty(WMQConstants.WMQ_MQMD_MESSAGE_CONTEXT,
  WMQConstants.WMQ_MDCTX_SET_IDENTITY_CONTEXT);

// On the message, set property to provide custom UserId
msg.setStringProperty("JMS_IBM_MQMD_UserIdentifier", "JoeBloggs");

// Send the message
// ...
```

JMS_IBM_MQMD_UserIdentifier' ı ayarlamadan önce WMQ_MQMD_MESSAGE_CONTEXT ayarını ayarlamanız gerekir. WMQ_MQMD_MESSAGE_CONTEXT kullanımı hakkında daha fazla bilgi için bkz. “JMS ileti nesnesi özellikleri” sayfa 862.

Benzer şekilde, bir ileti almadan önce WMQ_MQMD_READ_ENABLED ' i true değerine ayarlayarak ve daha sonra iletinin alma yöntemlerini (getStringözelliği gibi) kullanarak MQMD alanlarının içeriğini alabilirsiniz. Alınan özellikler salt okunurdur.

Bu örnek, bir iletinin MQMD.ApplIdentityData alanının değerini, bir kuyruktan ya da bir konudan elde edilen *değer* alanı ile sonuçlanır.

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(WMQConstants.WMQ_MQMD_READ_ENABLED, true);

// Receive a message
// ...

// Get MQMD field value using a property
String value = rcvMsg.getStringProperty("JMS_IBM_MQMD_ApplIdentityData");
```

JMS hedef nesne özellikleri

Hedef nesnenin iki özelliği JMS 'den MQMD' ye erişimi ve bir üçüncü denetim iletisi bağlamını denetler.

Çizelge 124. Özellik adları ve açıklamaları

Özellik	**Kısa Biçim**	**Tanım**
WMQ_MQMD_WRITE_ENABLED	MDC	Bir JMS uygulamasının MQMD alanlarının değerlerini ayarlayıp ayarlamayacağı
WMQ_MQMD_READ_ENABLED	MDR	Bir JMS uygulamasının MQMD alanlarının değerlerini ayıklayıp çıkaramayabileceği

Çizelge 124. Özellik adları ve açıklamaları (devamı var)		
Özellik	**Kısa Biçim**	**Tanım**
WMQ_MQMD_MESSAGE_ BAğLAMı	MDCTX	JMS uygulaması tarafından ayarlanacak ileti bağlamı düzeyi. Uygulama, bu özelliğin yürürlüğe girmesi için uygun bağlam yetkilisiyle çalışıyor olmalıdır.

Çizelge 125. Özellik adları, değerler ve küme yöntemleri			
Özellik	**Denetim aracındaki geçerli değerler (koyu olarak varsayılan değerler)**	**Programlardaki geçerli değerler**	**Set yöntemi**
WMQ_MQMD_WRITE ENABLED	• **HAYIR** Tüm JMS_IBM_MQMD* özellikleri yoksayılır ve değerleri, temeldeki MQMD yapısıyla kopyalanmaz. • EVET JMS_IBM_MQMD* özellikleri işlendi. Değerleri, temeldeki MQMD yapısına kopyalanır.	• **Yanlış** • Doğru	setMQMDWriteEtkinleştirildi
WMQ_MQMD_READ _ENABLED	• **HAYIR** İletileri gönderirken, gönderilen bir iletteki JMS_IBM_MQMD* özellikleri, MQMD ' deki güncellenen alan değerlerini yansıtacak şekilde güncellenmez. İleti alınırken, gönderenin bir kısmını ya da tümünü ayarlasa bile, alınan bir iletide JMS_IBM_MQMD* özelliklerinin hiçbiri kullanılabilir değil. • EVET İletileri gönderirken, gönderilen bir iletteki tüm JMS_IBM_MQMD* özellikleri, gönderenin belirtik olarak ayarlanmamış olması da içinde olmak üzere, MQMD ' deki güncellenmiş alan değerlerini yansıtacak şekilde güncellenir. İleti alınırken, tüm JMS_IBM_MQMD* özellikleri, gönderenin belirtik olarak ayarlamayanlar da içinde olmak üzere, alınan bir iletilerde kullanılabilir.	• **Yanlış** • Doğru	setMQMDReadEtkinleştirildi

Çizelge 125. Özellik adları, değerler ve küme yöntemleri (devamı var)			
Özellik	**Denetim aracındaki geçerli değerler (koyu olarak varsayılan değerler)**	**Programlardaki geçerli değerler**	**Set yöntemi**
WMQ_MQMD _MESSAGE_CONTEXT	• **VARSAYILAN** MQOPER API çağrısı ve MQPMO yapısı belirtik ileti bağlamı seçeneklerini belirtmiyor • SET_IDENTITY_CONTEXT MQOPEN API çağrısı, MQOO_SET_IDENTITY_CONTEXT ileti bağlamı seçeneğini belirtir ve MQPMO yapısı MQPMO_SET_IDENTITY_CONTEXT belirtisini belirtir • SET_ALL_CONTEXT MQOPEP API çağrısı, MQOO_SET_ALL_CONTEXT ileti bağlamı seçeneğini belirtir ve MQPMO yapısı MQPMO_SET_ALL_CONTEXT ' yi belirtir	• **WMQ_MDCTX_VARSAYıLAN** • WMQ_MDCTX_SET_IDENTITY_CONTEXT • WMQ_MDCTX_SET_ALL_CONTEXT	setMQMDMessageBağlam

JMS ileti nesnesi özellikleri
İleti nesnesi özellikleri önekli JMS_IBM_MQMD, ilgili MQMD alanını ayarlamanıza ya da okumanıza olanak sağlar.

İletilerin gönderilmesi

StrucId ve Version dışındaki tüm MQMD alanları temsil edilir. Bu özellikler yalnızca MQMD alanlarına gönderme yapar; hem MQMD ' de hem de MQRFH2 üstbilgisinde bir özellik oluşur, MQRFH2 içindeki sürüm belirlenmez ya da çıkarılmaz.

JMS_IBM_MQMD_BackoutCountdışında, bu özelliklerden herhangi biri ayarlanabilir. JMS_IBM_MQMD_BackoutCount için ayarlanan herhangi bir değer yok sayılır.

Bir özelliğin uzunluk üst sınırı varsa ve siz çok uzun bir değer sağlıyorsa, değer kesilir.

Bazı özellikler için, Hedef nesnede WMQ_MQMD_MESSAGE_CONTEXT özelliğini de ayarlamanız gerekir. Uygulamanın, bu özelliğin geçerli olması için uygun bağlam yetkisi ile çalışıyor olması gerekir. WMQ_MQMD_MESSAGE_CONTEXT değerini uygun bir değere ayarlamadıysanız, özellik değeri yoksayılır. WMQ_MQMD_MESSAGE_CONTEXT uygun bir değere ayarlıysa, ancak kuyruk yöneticisi için yeterli bağlam yetkisine sahip değilseniz, JMSException yayınlanır. WMQ_MQMD_MESSAGE_CONTEXT ile ilgili belirli değerleri gerektiren özellikler aşağıdaki gibidir.

Aşağıdaki özellikler WMQ_MQMD_MESSAGE_CONTEXT ' in WMQ_MDCTX_SET_IDENTITY_CONTEXT ya da WMQ_MDCTX_SET_ALL_CONTEXT olarak ayarlanmasını gerektirir:

- JMS_IBM_MQMD_UserIdentifier
- JMS_IBM_MQMD_AccountingToken
- JMS_IBM_MQMD_ApplIdentityVerileri

Aşağıdaki özellikler WMQ_MQMD_MESSAGE_CONTEXT ' in WMQ_MDCTX_SET_ALL_CONTEXT olarak ayarlanmasını gerektirir:

- JMS_IBM_MQMD_PutApplTipi

- JMS_IBM_MQMD_PutApplAdı
- JMS_IBM_MQMD_PutDate
- JMS_IBM_MQMD_PutTime
- JMS_IBM_MQMD_ApplOriginVerileri

İletileri alma

WMQ_MQMD_READ_ENABLED özelliği true değerine ayarlıysa, üreten uygulamanın ayarlı olduğu gerçek özelliklerden bağımsız olarak tüm bu özellikler alındı iletisi üzerinde kullanılabilir. Bir uygulama, ilk olarak JMS belirtimine göre tüm özellikler temizlenmedikçe, alınan iletinin özelliklerini değiştiremez. Alınan ileti, özellikler değiştirilmeden iletilebilir.

Uyarı: Uygulamanız WMQ_MQMD_READ_ENABLED özelliği true değerine ayarlanmış bir hedeften bir ileti alırsa ve onu WMQ_MQMD_WRITE_ENABLED değerine ayarlanmış bir hedefe iletirse, alınan iletinin tüm MQMD alanı değerleri, iletilen iletiye kopyalanmakta olan tüm MQMD alan değerlerinde sonuçlanır.

Özellikler tablosu

Bu çizelge, MQMD alanlarını temsil eden ileti nesnesinin özelliklerini listeler. Alanların ve izin verilen değerlerinin tam açıklamaları için bağlantılara bakın.

Çizelge 126. Özellik adları, açıklamalar ve tipler

Özellik	Tanım	Java Tipi	Tam açıklamayla bağlantı
JMS_IBM_MQMD_Report	Rapor iletileri için seçenekler	Tamsayı	Rapor
JMS_IBM_MQMD_MsgType	İleti tipi	Tamsayı	MsgType
JMS_IBM_MQMD_Expiry	İleti kullanım süresi	Tamsayı	Son kullanma tarihi
JMS_IBM_MQMD_Feedback	Geribildirim ya da neden kodu	Tamsayı	Geribildirim
JMS_IBM_MQMD_Kodlaması	İleti verilerinin sayısal kodlaması	Tamsayı	Kodlama
JMS_IBM_MQMD_CodedCharSetId	İleti verilerinin karakter kümesi tanıtıcısı	Tamsayı	CodedCharSetId
JMS_IBM_MQMD_Format	İleti verilerinin adını biçimle	Dizgi	Biçim
JMS_IBM_MQMD_Priority [1]	İleti önceliği	Tamsayı	Öncelik
JMS_IBM_MQMD_Persistence	İleti kalıcılığı	Tamsayı	Kalıcılık
JMS_IBM_MQMD_MsgId [2]	İleti Tanıtıcısı	Nesne (byte []) [4]	MsgId
JMS_IBM_MQMD_CorrelId [3]	İlinti tanıtıcısı	Nesne (byte []) [4]	CorrelId
JMS_IBM_MQMD_BackoutCount	Geriletme sayacı	Tamsayı	BackoutCount
JMS_IBM_MQMD_ReplyToQ	Yanıt kuyruğunun adı	Dizgi	ReplyToQ
JMS_IBM_MQMD_ReplyToQMgr	Yanıt kuyruğu yöneticisinin adı	Dizgi	ReplyToQMgr
JMS_IBM_MQMD_UserIdentifier	Kullanıcı kimliği	Dizgi	UserIdentifier

Çizelge 126. Özellik adları, açıklamalar ve tipler (devamı var)

Özellik	Tanım	Java Tipi	Tam açıklamayla bağlantı
JMS_IBM_MQMD_AccountingToken	Hesap simgesi	Nesne (byte []) [4]	AccountingToken
JMS_IBM_MQMD_ApplIdentityVerileri	Kimlikle ilgili uygulama verileri	Dizgi	ApplIdentityVerileri
JMS_IBM_MQMD_PutApplTipi	İletiyi koyan uygulamanın tipi	Tamsayı	PutApplTipi
JMS_IBM_MQMD_PutApplAdı	İletiyi koyan uygulamanın adı	Dizgi	PutApplAdı
JMS_IBM_MQMD_PutDate	İletinin konulduğu tarih	Dizgi	PutDate
JMS_IBM_MQMD_PutTime	İletinin konulduğu saat	Dizgi	PutTime
JMS_IBM_MQMD_ApplOriginVerileri	Köken ile ilgili uygulama verileri	Dizgi	ApplOriginVerileri
JMS_IBM_MQMD_GroupId	Grup tanıtıcısı	Nesne (byte []) [4]	GroupId
JMS_IBM_MQMD_MsgSeqNumarası	Grup içindeki mantıksal iletinin sıra numarası	Tamsayı	MsgSeqNumarası
JMS_IBM_MQMD_Göreli Konum	Mantıksal iletinin başlangıcındaki fiziksel iletilerde verilerin göreli konumu	Tamsayı	Göreli Konum
JMS_IBM_MQMD_MsgFlags	İleti İşaretleri	Tamsayı	MsgFlags
JMS_IBM_MQMD_OriginalLength	Özgün iletinin uzunluğu	Tamsayı	OriginalLength

1. **Uyarı:** JMS_IBM_MQMD_Priority değerine 0-9 aralığı içinde olmayan bir değer atarsanız, JMS belirtimini ihlal eder.
2. **Uyarı:** JMS belirtimi, ileti tanıtıcısının JMS sağlayıcısı tarafından ayarlanması gerektiğini ve bunun benzersiz ya da boş değer olması gerektiğini belirtiyor. JMS_IBM_MQMD_MsgIddeğerine bir değer atarsanız, bu değer JMSMessageID' ye kopyalanır. Bu nedenle JMS sağlayıcısı tarafından ayarlanmamış ve benzersiz olmayabilir: bu, JMS belirtimini ihlal eder.
3. **Uyarı:** JMS_IBM_MQMD_CorrelId değerine, 'ID:' dizgisiyle başlayan bir değer atarsanız, bu, JMS belirtimini ihlal eder.
4. **Uyarı:** Bir iletideki bayt dizisi özelliklerinin kullanımı, JMS belirtimini ihlal eder.

JMS için WebSphere MQ sınıflarını kullanarak bir uygulamadan IBM WebSphere MQ ileti verilerine erişilmesi

JMS için IBM WebSphere MQ sınıflarını kullanarak bir uygulama içinde eksiksiz WebSphere MQ ileti verilerine erişebilirsiniz. Tüm verilere erişmek için, iletinin bir `JMSBytesMessage`olması gerekir. `JMSBytesMessage` gövdesinde herhangi bir `MQRFH2` üstbilgisi, diğer IBM WebSphere MQ üstbilgileri ve aşağıdaki ileti verileri bulunur.

Set the `WMQ_MESSAGE_BODY` property of the destination to `WMQ_MESSAGE_BODY_MQ`, to receive all the message body data in the `JMSBytesMessage`.

WMQ_MESSAGE_BODY , WMQ_MESSAGE_BODY_JMS ya da WMQ_MESSAGE_BODY_UNSPECIFIEDolarak ayarlanırsa, ileti gövdesi JMS MQRFH2 üstbilgisi olmadan döndürülür ve JMSBytesMessage ' ın özellikleri, RFH2içinde ayarlanan özellikleri yansıtır.

Bazı uygulamalar bu konuda açıklanan işlevleri kullanamıyor. If an application is connected to a WebSphere MQ V6 queue manager, or if it has set SAğLAMA Sü to 6, the functions are not available.

İleti gönderilmesi

İletiler gönderilirken hedef özellik, WMQ_MESSAGE_BODY, WMQ_TARGET_CLIENT' a göre öncelik kazanır.

WMQ_MESSAGE_BODY , WMQ_MESSAGE_BODY_JMSolarak ayarlanırsa, JMS için WebSphere MQ sınıfları otomatik olarak JMSMessage özellikleri ve üstbilgi alanları ayarlarına dayalı olarak bir MQRFH2 üstbilgisi oluşturur.

WMQ_MESSAGE_BODY seçeneği WMQ_MESSAGE_BODY_MQolarak ayarlandıysa, ileti gövdesine ek üstbilgi eklenmez.

WMQ_MESSAGE_BODY , WMQ_MESSAGE_BODY_UNSPECIFIEDolarak ayarlandıysa, WMQ_TARGET_CLIENT değeri WMQ_TARGET_DEST_MQolarak ayarlandıysa, JMS için WebSphere MQ sınıfları bir MQRFH2 üstbilgisi gönderir. On receive, setting WMQ_TARGET_CLIENT to WMQ_TARGET_DEST_MQ results in any MQRFH2 being removed from the message body.

Not: JMSBytesMessage ve JMSTextMessage , JMSStreamMessage, JMSMapMessage,andve JMSObjectMessage gibi bir MQRFH2gerektirmez.

WMQ_MESSAGE_BODY_UNSPECIFIED , WMQ_MESSAGE_BODYiçin varsayılan ayardır ve WMQ_TARGET_DEST_JMS , WMQ_TARGET_CLIENTiçin varsayılan ayardır.

Bir JMSBytesMessagegönderdiğinizde, WebSphere MQ iletisi oluşturulduğunda JMS ileti gövdesine ilişkin varsayılan ayarları geçersiz kılabilirsiniz. Aşağıdaki özellikleri kullanın:

- JMS_IBM_Format ya da JMS_IBM_MQMD_Format: Bu özellik, önceki bir Websphere MQ üstbilgisi yoksa, JMS ileti gövdeini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün biçimini belirtir.
- JMS_IBM_Character_Set ya da JMS_IBM_MQMD_CodedCharSetId: Bu özellik, önceki Websphere MQ üstbilgisi yoksa, JMS ileti gövdenini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün CCSID değerini belirtir.
- JMS_IBM_Encoding ya da JMS_IBM_MQMD_Encoding: Bu özellik, önceki bir Websphere MQ üstbilgisi yoksa, JMS ileti gövdeini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün kodlamasını belirtir.

If both types of property are specified, the JMS_IBM_MQMD_* properties override the corresponding JMS_IBM_* properties, as long as the destination property WMQ_MQMD_WRITE_ENABLED is set to true.

JMS_IBM_MQMD_* ve JMS_IBM_* kullanılarak ileti özellikleri ayarı arasında yürürlükte olan farklar önemlidir:

1. JMS_IBM_MQMD_* özellikleri, IBM WebSphere MQ JMS sağlayıcısına özgüledir.
2. JMS_IBM_MQMD_* özellikleri yalnızca MQMDiçinde ayarlanır. JMS_IBM_* properties are set in the MQMD only if the message does not have an MQRFH2 JMS header. Tersi durumda, JMS RFH2 üstbilgisinde ayarlanır.
3. JMS_IBM_MQMD_* özelliklerinin, bir JMSMessageiçine yazılan metin ve sayıların kodlamasını etkilemez.

   Alma uygulaması büyük olasılıkla, MQMD.Encoding ve MQMD.CodedCharSetId değerlerinin, ileti gövdesindeki sayı ve metin kümesi kodlamasına ve karakter kümesine karşılık geldiğini varsayar. JMS_IBM_MQMD_* özellikleri kullanılırsa, bunu yapmak için gönderme uygulamasının sorumluluğunda olur. İleti gövdesindeki sayı ve metin kodlaması ve karakter kümesi, JMS_IBM_* özellikleri tarafından ayarlanır.

   The badly coded snippet in Şekil 163 sayfa 866 sends a message encoded in character set 1208, with MQMD.CodedCharSetId set to 37.

a. Yanlış kodlanmış iletiyi gönder

```
TextMessage tmo = session.createTextMessage();
((MQDestination) destination).setMessageBodyStyle
                (WMQConstants.WMQ_MESSAGE_BODY_MQ);
((MQDestination)destination).setMQMDWriteEnabled(true);
tmo.setIntProperty(WMQConstants.JMS_IBM_MQMD_CODEDCHARSETID, 37);
tmo.setIntProperty(WMQConstants.JMS_IBM_CHARACTER_SET, 1208);
tmo.setText("String one");
producer.send(tmo);
```

b. Receiving the message, relying on the value of JMS_IBM_XX_ENCODE_CASE_ONE charter_set set by the value of MQMD.CodedCharSetId:

```
TextMessage tmi = (TextMessage) cons.receive();
System.out.println("Message is \"" + tmi.getText() + "\"");
```

c. Sonuç çıkışı:

```
Message is "ëÈÊ'>...??>?"
```

Şekil 163. Sürekli olarak kodlanan MQMD ve ileti verileri

Şekil 164 sayfa 866 içindeki kodun parçacıklarından biri, otomatik olarak oluşturulan bir MQRFH2 üstbilgisi eklenmeden, uygulamanın bilgi yükünü içeren gövdesi ile bir kuyruğa ya da konuya eklenmekte olan bir iletiyle sonuçlanıyor.

1. Setting WMQ_MESSAGE_BODY_MQ:

```
((MQDestination) destination).setMessageBodyStyle
                (WMQConstants.WMQ_MESSAGE_BODY_MQ);
```

2. Setting WMQ_TARGET_DEST_MQ:

```
((MQDestination) destination).setMessageBodyStyle
                (WMQConstants.WMQ_MESSAGE_BODY_UNSPECIFIED);
((MQDestination) destination).
                setTargetClient(WMQConstants.WMQ_TARGET_DEST_MQ);
```

Şekil 164. MQ ileti gövdesiyle bir ileti gönderin.

İleti alma

WMQ_MESSAGE_BODY seçeneği WMQ_MESSAGE_BODY_JMSolarak ayarlandıysa, gelen JMS ileti tipi ve gövdesi, alınan Websphere MQ iletisinin içeriğiyle belirlenir. The message type and body are determined by fields in the MQRFH2 header, or in the MQMD, if there is no MQRFH2.

WMQ_MESSAGE_BODY , WMQ_MESSAGE_BODY_MQolarak ayarlandıysa, gelen JMS ileti tipi JMSBytesMessageolur. JMS ileti gövdesi, temeldeki MQGET API çağrısı tarafından döndürülen ileti verileridir. İleti gövdesinin uzunluğu, MQGET çağrısı tarafından döndürülen uzunluktır. İleti gövdesindeki verilerin karakter takımı ve kodlaması, MQMD' un CodedCharSetId ve Kodlama alanları tarafından belirlenir. İleti gövdesindeki verilerin biçimi, MQMD' un Biçim alanı tarafından belirlenir.

WMQ_MESSAGE_BODY , WMQ_MESSAGE_BODY_UNSPECIFIEDdeğerine ayarlıysa, varsayılan değer, JMS için IBM WebSphere MQ sınıfları bunu WMQ_MESSAGE_BODY_JMSolarak ayarlar.

Bir JMSBytesMessagealdığınızda, aşağıdaki özelliklere başvurarak kodu çözebilirsiniz:

- JMS_IBM_Format ya da JMS_IBM_MQMD_Format: Bu özellik, önceki bir Websphere MQ üstbilgisi yoksa, JMS ileti gövdeini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün biçimini belirtir.

- `JMS_IBM_Character_Set` ya da `JMS_IBM_MQMD_CodedCharSetId`: Bu özellik, önceki Websphere MQ üstbilgisi yoksa, JMS ileti gövdenini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün `CCSID` değerini belirtir.
- `JMS_IBM_Encoding` ya da `JMS_IBM_MQMD_Encoding`: Bu özellik, önceki bir Websphere MQ üstbilgisi yoksa, JMS ileti gövdeini başlatan WebSphere MQ üstbilgisinin ya da uygulama bilgi yükünün kodlamasını belirtir.

The following code snippet results in a received message that is a `JMSBytesMessage`. Alınan iletinin içeriğinden ve alınan `MQMD`' ın biçim alanının içeriğinden bağımsız olarak, ileti bir `JMSBytesMessage`iletidir.

```
((MQDestination)destination).setMessageBodyStyle
                            (WMQConstants.WMQ_MESSAGE_BODY_MQ);
```

Hedef özelliği WMQ_MESSAGE_BODY
WMQ_MESSAGE_BODY, JMS uygulamasının bir WebSphere MQ iletisine ilişkin MQRFH2 ' yi ileti bilgi yükünün bir parçası olarak (yani JMS ileti gövdesinin bir parçası olarak) işleyip işlemediğini belirler.

Çizelge 127. Özellik adları ve açıklamaları

Özellik	**Kısa Biçim**	**Tanım**
WMQ_MESSAGE_BODY	GÖVDE	Bir JMS uygulamasının, ileti bilgi yükünün (JMS ileti gövdesinin bir parçası olarak) bir WebSphere MQ iletisine ilişkin MQRFH2 ' yi işleyip işlemediğini.

Çizelge 128. Özellik adları, değerler ve küme yöntemleri

Özellik	Denetim aracındaki geçerli değerler (koyu olarak varsayılan değerler)	Programlardaki geçerli değerler	Set yöntemi
WMQ_MESSAGE_BODY	• **Belirtilmedi** When sending, WebSphere MQ classes for JMS does or does not generate and include an MQRFH2 header, depending on the value of WMQ_TARGET_CLIENT. Alma sırasında, değer JMS olarak işlev görür. • JMS Gönderirken, JMS için WebSphere MQ sınıfları otomatik olarak bir MQRFH2 üstbilgisi oluşturur ve bunu WebSphere MQ iletisinde içerir. JMS için WebSphere MQ sınıfları alınırken, JMS ileti özelliklerini MQRFH2 'deki değerlere (varsa) göre ayarlayın; JMS ileti gövdesinin bir parçası olarak MQRFH2 ' yi göstermiyor. • MQ Gönderirken, JMS için WebSphere MQ sınıfları bir MQRFH2oluşturmaz. JMS için WebSphere MQ sınıfları alınırken, JMS ileti gövdesinin bir parçası olarak MQRFH2 ' yi sunar.	• **WMQ_MESSAGE_BODY_BELIRTILMEYEN** • WMQ_MESSAGE_BODY_JMS • WMQ_MESSAGE_BODY_MQ	setMessageBodyStyle

JMS kalıcı iletileri

JMS uygulamaları için WebSphere MQ sınıfları, bazı güvenilirlik pahasına, JMS kalıcı iletileri için daha iyi başarım sağlamak için **NonPersistentMessageClass** kuyruk özniteliğini kullanabilir.

Bir WebSphere MQ kuyruğu **NonPersistentMessageClass**adlı bir özniteliğe sahiptir. Bu özniteliğin değeri, kuyruk yöneticisi yeniden başlatıldığında kuyruktaki kalıcı olmayan iletilerin atılıp atılmayacağını belirler.

You can set the attribute for a local queue by using the WebSphere MQ Script (MQSC) command, DEFINE QLOCAL, with either of the following parameters:

NPMSıNıFı (NORMAL)
Kuyruk yöneticisi yeniden başlatıldığında, kuyruklardaki kalıcı olmayan iletiler atılır. Bu varsayılan değerdir.

NPMCLASS (YüKSEK)
Kuyruk yöneticisi susturulmuş ya da hemen sona erdirildiğinde kuyruk yöneticisi yeniden başlatıldığında, kuyruklardaki kalıcı olmayan iletiler atılmaz. Ancak, önleyici olmayan iletiler atılabilir; ancak, önleyici bir sona erdirme ya da hata nedeniyle atılabilir.

Bu konuda, JMS uygulamalarının WebSphere MQ sınıflarının JMS kalıcı iletileri için daha iyi başarım sağlamak amacıyla bu kuyruk özniteliğini nasıl kullanabilecekleri ele alınmıştır.

Bir Kuyruk ya da Konu nesnesinin PERSISTENCE özelliği HIGH değerine sahip olabilir. Bu değeri ayarlamak için WebSphere MQ JMS yönetim aracını kullanabilir ya da bir uygulama değiştirge olarak WMQConstants.WMQ_PER_NPHIGH değerini geçen Destination.setPersistence() yöntemini çağırabilir.

Bir uygulama, PERSISTENCE özelliğinin HIGH değeri olan bir hedefe JMS kalıcı iletisi ya da JMS kalıcı olmayan iletisi gönderirse ve temeldeki WebSphere MQ kuyruğu NPMCLASS (HIGH) olarak ayarlandıysa, ileti kuyruğa WebSphere MQ kalıcı olmayan iletisi olarak gönderilir. Hedefin PERSISTENCE özelliğinin HIGH değeri yoksa ya da temeldeki kuyruk npmclass (normal) olarak ayarlandıysa, kuyruğa bir WebSphere MQ kalıcı iletisi olarak bir JMS kalıcı iletisi konursa ve bir JMS kalıcı olmayan iletisi, kuyruğa WebSphere MQ kalıcı olmayan bir ileti olarak konursa, kuyruğa kalıcı bir ileti yerleştirilir.

Bir JMS kalıcı iletisi bir kuyruğa WebSphere MQ kalıcı olmayan iletisi olarak konursa ve bir kuyruk yöneticisinin susturulmuş ya da hemen kapanması sonrasında iletinin atılmamasını sağlamak istiyorsanız, iletinin yöneltileceği tüm kuyruklar NPMCLASS (HIGH) olarak ayarlanmalıdır. Yayınlama/abone olma etki alanında bu kuyruklar, abone kuyruklarını içerir. As an aid to enforcing this configuration, WebSphere MQ classes for JMS throws an InvalidDestinationException if an application tries to create a message consumer for a destination where the PERSISTENCE property has the value HIGH and the underlying WebSphere MQ queue is set to NPMCLASS(NORMAL).

Bir hedefin HIGH ' a PERSISTENCE özelliğinin ayarlanması, o hedeften alınan bir iletinin nasıl alınmadığını etkilemez. JMS kalıcı iletisi olarak JMS kalıcı iletisi olarak gönderilen bir ileti alındı ve JMS kalıcı olmayan iletisi olarak gönderilen bir ileti JMS kalıcı olmayan iletisi olarak alındı.

Bir uygulama, PERSISTENCE özelliğinin HIGH değerine sahip olduğu bir hedefe ilk iletiyi gönderdiğinde ya da bir uygulama, PERSISTENCE özelliğinin HIGH, WebSphere MQ sınıflarında NPMCLASS (YÜKSEK) değerinin temeldeki WebSphere MQ kuyruğunda ayarlanıp ayarlanmadığını saptamak için bir MQINQ çağrısı içeren bir hedef için ilk ileti tüketicisi yarattığında, bir uygulama ilk ileti tüketicisi yaratır. Bu nedenle, uygulamanın kuyruğa sorgulama yetkisi olmalıdır. Buna ek olarak, JMS için WebSphere MQ sınıfları, hedef silininceye kadar MQINQ çağrısının sonucunu korur ve daha fazla MQINQ çağrısı yayınlamaz. Bu nedenle, uygulama hala hedef kullanılırken, temeldeki kuyruğun NPMCLASSS ayarını değiştirirseniz, JMS için WebSphere MQ sınıfları yeni ayarı fark etmez.

JMS kalıcı iletilerinin WebSphere MQ kuyruklarını WebSphere MQ kalıcı olmayan iletileri olarak yerleştirmesine izin vererek, bazı güvenilirlik giderlerinde performans kazanıyorsunuz. JMS kalıcı iletileri için maksimum güvenilirlik gereksiniminiz varsa, iletileri PERSISTENCE özelliğinin HIGH değerine sahip olduğu bir hedefe göndermeyin.

JMS Katmanı, SYSTEM.DEFAULT.MODEL.QUEUEyerine SYSTEM.JMS.TEMPQ.MODELkullanabilir. SYSTEM.JMS.TEMPQ.MODEL , kalıcı iletileri kabul edemeyen kalıcı dinamik kuyruklar oluşturur; SYSTEM.DEFAULT.MODEL.QUEUE kalıcı iletileri kabul edemez. Kalıcı iletileri kabul etmek için geçici kuyruklar kullanmak istiyorsanız, bu nedenle SYSTEM.JMS.TEMPQ.MODEL(MODEL) ya da model kuyruğunu, seçtiğiniz alternatif bir kuyruğa çevirin.

JMS için WebSphere MQ sınıflarıyla SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) kullanılıyor

JMS uygulamaları için WebSphere MQ sınıfları SSL şifrelemesini kullanabilir. Bunu yapmak için bir JSSE sağlayıcısına gereksinim duyarlar.

WebSphere MQ classes for JMS connections using TRANSPORT(CLIENT) support Secure Sockets Layer (SSL) encryption. SSL, iletişim şifrelemesi, kimlik doğrulaması ve ileti bütünlüğü sağlar. Bu, genellikle İnternet üzerinde ya da bir intranet içinde herhangi iki eş arasında iletişim sağlamak için kullanılır.

JMS için WebSphere MQ sınıfları, SSL şifrelemesini işlemek için Java Secure Socket Extension (JSSE) kullanır ve bu nedenle bir JSSE sağlayıcısı gerektirir. JSE v1.4 JVM ' ler yerleşik bir JSSE sağlayıcısına

sahiptir. Sertifikaların nasıl yönetileceği ve saklanabileceği ile ilgili ayrıntılar sağlayıcıdan sağlayıcıya göre değişebilir. Bu konuda bilgi almak için JSSE sağlayıcısının belgelerine bakın.

Bu bölümde, JSSE sağlayıcınızın doğru bir şekilde kurulmuş ve yapılandırılmış olduğu ve JSSE sağlayıcınız için uygun sertifikaların kurulmuş ve kullanılabilir durumda olduğu varsayılır. Artık bir dizi yönetim özelliği ayarlamak için JMSAdmin 'i kullanabilirsiniz.

JMS uygulamasına ilişkin WebSphere MQ sınıflarınız bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, “JMS için IBM WebSphere MQ classes for JMS” sayfa 878konusuna bakın.

SSLCIPHERSUIT nesne özelliği
Bir ConnectionFactory nesnesi üzerinde SSL şifrelemesini etkinleştirmek için SSLCIPHERSUIT öğesini ayarlayın.

Bir ConnectionFactory nesnesi üzerinde SSL şifrelemesini etkinleştirmek için, SSLCIPHERSUIT özelliğini JSSE sağlayıcınız tarafından desteklenen bir CipherSuite olarak ayarlamak için JMSAdmin 'i kullanın. Bu, hedef kanaldaki CipherSpec ayarına uygun olmalıdır. Ancak, CipherSuites , CipherSpecs ' ten farklıdır ve bu nedenle farklı adlara sahiptir. “JMS ' de SSL CipherSpecs ve CipherSuites” sayfa 873 contains a table mapping the CipherSpecs supported by WebSphere MQ to their equivalent CipherSuites as known to JSSE. For more information about CipherSpecs and CipherSuites with WebSphere MQ, see Durumu.

For example, to set up a ConnectionFactory object that can be used to create a connection over an SSL enabled MQI channel with a CipherSpec of RC4_MD5_EXPORT, issue the following command to JMSAdmin:

```
ALTER CF(my.cf) SSLCIPHERSUITE(SSL_RSA_EXPORT_WITH_RC4_40_MD5)
```

This can also be set from an application, using the setSSLCipherSuite() method on an MQConnectionFactory object.

Kolaylık sağlamak amacıyla, SSLCIPHERSUITE özelliğinde bir CipherSpec belirtilirse, JMSAdmin CipherSpec ' yi uygun bir CipherSuite ile eşleştirmeyi dener ve bir uyarı yayınlar. Özellik bir uygulama tarafından belirtilmişse, bu eşleme girişimi yapılmaz.

CCDT (Client Channel Definition Table; İstemci Kanal Tanımlama Çizelgesi) olanağını kullanın. Daha fazla bilgi için “JMS için IBM WebSphere MQ classes for JMS” sayfa 878 başlıklı konuya bakın.

SSLFIPSREQUIRD nesne özelliği
IBM Java JSSE FIPS sağlayıcısı (IBMJSSEFIPS) tarafından desteklenen bir CipherSuite özelliğini kullanmak için bağlantı gerekiyorsa, bağlantı üreticisinin SSLFIPSREQUIRD özelliğini YES değerine ayarlayın.

Bu özelliğin varsayılan değeri NO ise, bir bağlantının WebSphere MQtarafından desteklenen herhangi bir CipherSuite ' i kullanabileceği anlamına gelir.

Bir uygulama birden çok bağlantı kullanıyorsa, uygulama ilk bağlantıyı yarattığında kullanılan SSLFIPSIN değeri, uygulama birbirini izleyen herhangi bir bağlantı yarattığında kullanılan değeri belirler. Bu, sonraki bir bağlantı yaratmak için kullanılan bağlantı üreticisinin SSLFIPSREQUIRD özelliğinin değerinin dikkate alınmamasını gösterir. Farklı bir SSLFIPSREQUIRET değeri kullanmak istiyorsanız uygulamayı yeniden başlatmalısınız.

Bir uygulama, ConnectionFactory nesnesinin setSSLFipsRequired () yöntemini çağırarak bu özelliği ayarlayabilir. CipherSuite ayarı belirlenmezse, özellik yoksayılır.

İlgili görevler
MQI istemcisinde çalıştırma sırasında yalnızca FIPS onaylı CipherSpecs ' in kullanıldığını belirtme
İlgili başvurular
UNIX, Linux ve Windows için Federal Bilgi İşleme Standartları (FIPS)

SSLPEERNAME nesne özelliği
JMS uygulamanızın doğru kuyruk yöneticisine bağlanmasını sağlamak için ayırt edici ad örüntlerini belirtmek için SSLPEERNAME değerini kullanın.

Bir JMS uygulaması, ayırt edici ad (DN) kalıbı belirterek doğru kuyruk yöneticisine bağlanmasını sağlar. Bağlantı yalnızca, kuyruk yöneticisi örüntüyle eşleşen bir DN (DN) sunarsa başarılı olur. Bu örüntüm biçimiyle ilgili ayrıntılar için ilgili konulara bakın.

DN, bir ConnectionFactory nesnesine ilişkin SSLPEERNAME özelliği kullanılarak ayarlanır. Örneğin, aşağıdaki JMSAdmin komutu bir ConnectionFactory nesnesini, kuyruk yöneticisinin kendisini `QMGR.` karakterleriyle başlayan bir Ortak Ad ile ve en az iki Kuruluş Birimi adında IBM ve ikinci WEBSPHERE olmak üzere en az iki Kuruluş Birimi adıyla tanıtmasını beklemesini sağlar.

```
ALTER CF(my.cf) SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

Denetleme, büyük/küçük harfe duyarlı değildir ve virgül yerine noktalı virgüller yerleştirilebilir. SSLPEERNAME, bir MQConnectionFactory nesnesindeki setSSLPeerName () yöntemini kullanarak bir uygulamadan da ayarlanabilir. Bu özellik belirlenmezse, kuyruk yöneticisi tarafından sağlanan Ayırt Edici Ad üzerinde herhangi bir denetleme gerçekleştirilmez. CipherSuite ayarı belirlenmezse bu özellik yok sayılır.

SSLCERTSTORS nesne özelliği
Sertifika iptal listesi (CRL) denetimi için kullanılacak LDAP sunucularının bir listesini belirlemek için SSLCertstors öğesini kullanın.

Artık güvenilir olmayan sertifikaları tanımlamak için sertifika iptal listesi (CRL) kullanılması yaygındır. CRL ' ler genellikle LDAP sunucularında barındırılır. JMS, CRL denetimi için Java 2 v1.4 ya da üstü altında bir LDAP sunucusunun belirtilmesine olanak sağlar. Aşağıdaki JMSAdmin örneği, JMS ' yi crl1.ibm.com:adlı bir LDAP sunucusunda barındırılan bir CRL kullanmaya yönlendirir.

```
ALTER CF(my.cf) SSLCRL(ldap://crl1.ibm.com)
```

Not: Bir LDAP sunucusunda barındırılan bir CRL ile başarılı bir şekilde CertStore kullanmak için, Java Software Development Kit (SDK) sunucunuzun CRL ile uyumlu olduğundan emin olun. Bazı SDK 'lar, CRL' nin, LDAP v2için bir şema tanımlayan RFC 2587 'ye uymasını gerektirir. Çoğu LDAP v3 sunucusu, bunun yerine RFC 2256 'yı kullanır.

LDAP sunucunuz varsayılan 389 kapısında çalışmıyorsa, kapıyı iki nokta üst üste (:) ve kapı numarasını anasistem adına ekleyerek belirleyebilirsiniz. Kuyruk yöneticisi tarafından sunulan sertifika crl1.ibm.com'da bulunan CRL' de varsa, bağlantı tamamlanmamaktadır. Tek bir hata noktasından kaçınmak için JMS, birden çok LDAP sunucusunun, boşluk karakteriyle sınırlanmış LDAP sunucularının bir listesi sağlanarak sağlanmasını sağlar. Örnek:

```
ALTER CF(my.cf) SSLCRL(ldap://crl1.ibm.com ldap://crl2.ibm.com)
```

Birden çok LDAP sunucusu belirtildiğinde, JMS her birini, kuyruk yöneticisinin sertifikasını başarıyla doğrulayabilecek bir sunucu buluncaya kadar sırayla dener. Her bir sunucunun aynı bilgileri içermesi gerekir.

Bu biçimdeki bir dizgi, MQConnectionFactory.setSSLCertStores () yöntemindeki bir uygulama tarafından sağlanabilir. Diğer bir seçenek olarak, uygulama bir ya da daha çok java.security.cert.CertStore nesnesi oluşturabilir, bunları uygun bir Toplama nesnesine yerleştirebilir ve bu Veri Toplama nesnesini setSSLCertStores () yöntemine sağlayabilir. Bu şekilde, uygulama CRL denetimini özelleştirebilir. CertStore nesnelerini oluşturmaya ve kullanmaya ilişkin ayrıntılar için JSSE belgelerinize bakın.

Bir bağlantı ayarlandığında, kuyruk yöneticisi tarafından sunulan sertifika aşağıdaki gibi doğrulanır:

1. Kaynak grubundaki ilk CertStore nesnesi, bir CRL sunucusunu tanımlamak için kullanılır. sslCertdepolar tarafından tanımlanır.
2. CRL sunucusuyla bağlantı kurma girişiminde bulunmanız gerekir.

3. Girişim başarılı olursa, sunucu, sertifikan için bir eşleşme için arama yapılır.
   a. Sertifikana geri alınacak bir sertifika bulunursa, arama işlemi sona erir ve bağlantı isteği, MQRC_SSL_CERTIFICATE_FESHRIVE neden kodlarıyla başarısız olur.
   b. Sertifika bulunamazsa, arama işlemi sona ermiş ve bağlantının devam etmesine izin verilir.
4. Sunucuyla iletişim kurma girişimi başarısız olursa, bir sonraki CertStore nesnesi bir CRL sunucusunu tanımlamak için kullanılır ve süreç 2. adımdan yinelenir.

   Bu, derlemdeki son CertStore ise ya da Kaynak Grubu hiçbir CertStore nesnesi içermiyorsa, arama işlemi başarısız olur ve bağlantı isteği MQRC_SSL_CERT_STORE_ERROR neden koduyla başarısız olur.

Veri Toplama nesnesi, CertStores ' un kullanıldığı sırayı belirler.

Uygulamanız bir CertStore nesneleri derlemi ayarlamak için setSSLCertStores () kullanıyorsa, MQConnectionFactory artık bir JNDI ad alanına bağ tanımlanmaz. Bunu gerçekleştirme girişimi bir kural dışı duruma neden olur. sslCertStores özelliği ayarlanmazsa, kuyruk yöneticisi tarafından sağlanan sertifikada herhangi bir iptal denetimi gerçekleştirilmez. CipherSuite ayarı belirlenmezse bu özellik yok sayılır.

SSSRESETCOUNT nesne özelliği
Bu özellik, şifreleme için kullanılan gizli anahtardan önce bir bağlantı tarafından gönderilen ve alınan toplam bayt sayısını temsil eder.

Gönderilen bayt sayısı, şifrelemeden önceki sayıdır ve alınan bayt sayısı, şifre çözme işleminden sonra gelen sayıdır. Bayt sayısı, JMS için WebSphere MQ sınıfları tarafından gönderilen ve alınan denetim bilgilerini de içerir.

Örneğin, 4 MB ' lik veri akıldıktan sonra yeniden anlaşma sağlanan bir SSL etkin MQI kanalı üzerinden bir bağlantı yaratmak üzere kullanılabilecek bir ConnectionFactory nesnesini yapılandırmak için, JMSAdmin komutunu aşağıdaki komutu verin:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

Bir uygulama, bir ConnectionFactory nesnesinin setSSLResetCount () yöntemini çağırarak bu özelliği ayarlayabilir.

Bu özelliğin değeri sıfır ise, varsayılan değer olan gizli anahtar hiçbir zaman yeniden anlaşmamalıdır. CipherSuite ayarı belirlenmezse, özellik yoksayılır.

SSLSocketFactory nesne özelliği
Bir uygulamaya ilişkin SSL bağlantısının diğer özelliklerini uyarlamak için bir SSLSocketFactory yaratın ve JMS 'yi kullanarak JMS' yi kullanacak şekilde yapılandırın.

Bir uygulama için SSL bağlantısının diğer yönlerini uyarlamak isteyebilirsiniz. Örneğin, şifreleme donanımını başlatmak ya da anahtar deposunu ve güvenli deponun kullanımını değiştirmek isteyebilirsiniz. Bunu yapmak için, uygulamanın öncelikle uygun şekilde özelleştirilen bir javax.net.ssl.SSLSocketFactory nesnesi yaratması gerekir. Özelleştirilebilir özellikler sağlayıcıdan sağlayıcıya farklılık gösterdiğinden, bunu nasıl yapacağa ilişkin bilgi için JSSE belgelerinize bakın. Uygun bir SSLSocketFactory nesnesi elde edildikten sonra, JMS ' yi özelleştirilmiş SSLSocketFactory nesnesini kullanacak şekilde yapılandırmak için MQConnectionFactory.setSSLSocketFactory () yöntemini kullanın.

Uygulamanız, özelleştirilmiş bir SSLSocketFactory nesnesi ayarlamak için setSSLSocketFactory () yöntemini kullanıyorsa, MQConnectionFactory nesnesi artık bir JNDI ad alanına bağlanmaz. Bunu gerçekleştirme girişimi bir kural dışı duruma neden olur. Bu " zellik belirlenmezse, varsaylan SSLSocketFactory nesnesi kullanlmr. Varsayılan SSLSocketFactory nesnesinin davranışına ilişkin ayrıntılar için JSSE belgelerinize bakın. CipherSuite ayarı belirlenmezse bu özellik yok sayılır.

Önemli: Güvenli olmayan bir JNDI ad alanından bir ConnectionFactory nesnesi alındığında SSL özelliklerinin kullanılmasının güvenlik güvenceye alınmasını güvenceye aldığından emin olun. Özellikle, JNDI ' nin standart LDAP somutlaması güvenli değildir. Bir saldırgan LDAP sunucusunu taklit edebilir, bir JMS uygulamasını fark etmeden yanlış sunucuya bağlanmaya yönlendirebilir. Uygun güvenlik düzenlemeleriyle, diğer JNDI somutlamaları (fscontext somutlaması gibi) güvenlidir.

JSSE anahtar deposunda ya da güvenilir depoda değişiklik yapılması
Anahtar deposunda ya da güvenilir depoda değişiklik yaparsanız, değişikliklerin alınabilmesine ilişkin bazı işlemleri yapmanız gerekir.

JSSE anahtar deposu ya da güvenilirlik deposunun içeriğini değiştirirseniz ya da anahtar deposunun ya da güvenilir depo dosyasının konumunu değiştirdiğinizde, o sırada çalışan JMS uygulamaları için WebSphere MQ sınıflarının otomatik olarak değişiklikleri almaması gerekir. Değişikliklerin yürürlüğe girmesi için aşağıdaki işlemler gerçekleştirilmelidir:

- Uygulamalar tüm bağlantılarını kapatmalı ve bağlantı havuzlarındaki kullanılmayan bağlantıları yok etmelidir.
- JSSE sağlayıcınız, bilgileri anahtar deposundan ve güvenilir depodan önbelleğe aldıysa, bu bilgiler yenilenmelidir.

Bu işlemler gerçekleştirildikten sonra, uygulamalar bağlantılarını yeniden yaratabilirler.

Uygulamalarınızı nasıl tasarlamanıza ve JSSE sağlayıcınız tarafından sağlanan işlevlere bağlı olarak, uygulamalarınızı durdurup yeniden başlatmaksızın bu işlemleri gerçekleştirmeniz de mümkün olabilir. Ancak, uygulamaların durdurulması ve yeniden başlatılması en basit çözüm olabilir.

JMS ' de SSL CipherSpecs ve CipherSuites
CipherSpecs , WebSphere MQ ve bunların eşdeğeri CipherSuitestarafından desteklenir.

Çizelge 129 sayfa 873 , WebSphere MQ ve eşdeğer CipherSuitestarafından desteklenen CipherSpecs ' i listeler. ConnectionFactory özelliği SSLFIPSREQUIRD değeri NO olarak ayarlanırsa, JMS uygulaması için bir WebSphere MQ sınıfları, MQI kanalının sunucu ucunda desteklenen bir CipherSpec belirtilirse ve istemci ucunda CipherSuite eşdeğeri belirtilirse, bir kuyruk yöneticisine bağlanabilir. SSLFIPSREQUIRD YES olarak ayarlandıysa, CipherSpec ve CipherSuite birleşimi, uygulamanın kuyruk yöneticisine bağlanıp bağlanmayacağını belirler.

Bir MQI kanalının sunucu ucunda, CipherSpec adı, DEFINE CHANNEL CHLTYPE (SVRCONN) komutunda SSLCIPH parametresinin değeri olarak belirlenebilir. Bir MQI kanalının istemci ucunda, bir CipherSuite adının adı aşağıdaki şekillerde belirtilebilir:

- Bir uygulama, ConnectionFactory nesnesinin setSSLCipherSuite () yöntemini çağırabilir.
- WebSphere MQ JMS yönetim aracı kullanılarak, bir ConnectionFactory nesnesinin SSLCIPHERSUIT özelliğini ayarlayabilirsiniz.

Çizelge 129. CipherSpecs , WebSphere MQ tarafından desteklenir ve eşdeğeri CipherSuites

CipherSpec	**Eşdeğer CipherSuite**	**SFIPS [1] YES değerine ayarlıysa bağlantı yapılabilir mi?**
NULL_MD5	SSL_RSA_WITH_NULL_MD5	Hayır
NULL_SHA	SSL_RSA_WITH_NULL_SHA	Hayır
RC4_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC4_40_MD5	Hayır
RC4_MD5_US	SSL_RSA_WITH_RC4_128_MD5	Hayır
RC4_SHA_US	SSL_RSA_WITH_RC4_128_SHA	Hayır
RC2_MD5_EXPORT	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5	Hayır
DESTE_SHA_EXPORT	SSL_RSA_WITH_DES_CBC_SHA	Hayır
RC4_56_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_RC4_56_SHA	Hayır
DES_SHA_EXPORT1024	SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA	Hayır

Çizelge 129. CipherSpecs , WebSphere MQ tarafından desteklenir ve eşdeğeri CipherSuites (devamı var)		
CipherSpec	**Eşdeğer CipherSuite**	**SFIPS [1] YES değerine ayarlıysa bağlantı yapılabilir mi?**
TRIPLE_DES_SHA_US	SSL_RSA_WITH_3DES_EDE_CBC_SHA	Hayır
TLS_RSA_WITH_NULL_SHA256	SSL_RSA_WITH_NULL_SHA256	Hayır [7]
TLS_RSA_WITH_AES_128_CBC_SHA	SSL_RSA_WITH_AES_128_CBC_SHA	Evet [5] [7]
TLS_RSA_WITH_AES_256_CBC_SHA	SSL_RSA_WITH_AES_256_CBC_SHA	Evet [5] [7]
TLS_RSA_WITH_AES_256_CBC_SHA256	SSL_RSA_WITH_AES_256_CBC_SHA256	Evet [5] [7]
AES_SHA_US[2]		
TLS_RSA_WITH_DES_CBC_SHA[8] [9]	SSL_RSA_WITH_DES_CBC_SHA	Hayır[3]
TLS_RSA_WITH_3DES_EDE_CBC_SHA[8]	SSL_RSA_WITH_3DES_EDE_CBC_SHA	Evet
FIPS_WITH_DES_CBC_SHA	SSL_RSA_FIPS_WITH_DES_CBC_SHA	Hayır[4]
FIPS_WITH_3DES_EDE_CBC_SHA	SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA	Hayır[6]

Notlar:

1. WebSphere MQ JMS yönetim aracı kullanılırken, SFIPS, ConnectionFactory özelliğinin kısa adıdır. SSLFIPSREQUIRD.
2. Bu CipherSpec , eşdeğer bir CipherSuite' e sahip değildir.
3. Bu CipherSpec , 19th Mayıs 2007 tarihinden önce FIPS 140-2 sertifikasıydı.
4. Bu CipherSpec , 19th Mayıs 2007 tarihinden önce FIPS 140-2 sertifikasıydı. FIPS_WIT_DES_CBC_SHA adı geçmiştir ve bu CipherSpec ' in daha önce olduğu (ancak artık değil) FIPS-uyumlu olduğu gerçeğini yansıtır. Bu CipherSpec kullanımdan kaldırıldı ve kullanımı önerilmiyor.
5. These CipherSpecs (TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA256) cannot be used to secure a connection from the WebSphere MQ Explorer to a queue manager unless the appropriate unrestricted policy files are applied to the JRE used by the Explorer.

   İlke dosyalarına ilişkin ek bilgi için Güvenlik bilgileri başlıklı konuya bakın.
6. The name FIPS_WITH_3DES_EDE_CBC_SHA is historical and reflects the fact that this CipherSpec was previously (but is no longer) FIPS-compliant. Bu CipherSpec kullanımdan kaldırıldı ve kullanımı önerilmiyor.
7. Bu CipherSpecs (TLS_RSA_WITH_NULL_SHA256, TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA256), IBM JREs 6.0 SR13 FP2 , 7.0 SR4 FP2 ya da sonraki bir sürümü gerektirir.
8. Bu CipherSpecs (TLS_RSA_WITH_3DES_EDE_CBC_SHA, TLS_RSA_WITH_DES_CBC_SHA, TLS_RSA_WITH_RC4_128_SHA256), SSLv3 ya da TLS ' yi kullanabilir. Varsayılan olarak, FIPS etkin olmadığı zaman SSLv3 kullanılır. TLS ' yi kullanmak için **com.ibm.mq.cfg.preferTLS** Java System Property özelliğini `true`olarak ayarlayın.
9. Bu CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA kullanımdan kaldırılmıştır. However, it can still be used to transfer up to 32 GB of data before the connection is terminated with error AMQ9288. Bu hatayı önlemek için, üçlü DES kullanmaktan kaçınmanız ya da bu CipherSpeckomutunu kullanırken gizli anahtar sıfırlamayı etkinleştirmeniz gerekir.

İlgili bilgiler

MQI istemcisinde çalıştırma sırasında yalnızca FIPS onaylı CipherSpecs ' in kullanıldığını belirtme

UNIX, Linux ve Windows için Federal Bilgi İşleme Standartları (FIPS)

JMS için WebSphere MQ sınıfları için Java 'da kanal çıkışları yazılıyor

Belirlenen arabirimleri uygulayan Java sınıflarını tanımlayarak kanal çıkışları yaratıyorsunuz.

com.ibm.mq.exits paketinde üç arabirim tanımlanır:

- Bir gönderme çıkışı içinWMQSendExit
- Bir alma çıkışı içinWMQReceiveExit
- WMQSecurityExit, for a security exit

Aşağıdaki örnek kod, üç arabirimi gerçekleştiren bir sınıfı tanımlıyor:

```
public class MyMQExits implements
WMQSendExit, WMQReceiveExit, WMQSecurityExit {
    // Default constructor
  public MyMQExits(){
  }
    // This method implements the send exit interface
  public ByteBuffer channelSendExit(
MQCXP channelExitParms,
                                  MQCD channelDefinition,
                                  ByteBuffer agentBuffer)
  {
    // Complete the body of the send exit here
  }
    // This method implements the receive exit interface
  public ByteBuffer channelReceiveExit(
MQCXP channelExitParms,
                                     MQCD channelDefinition,
                                     ByteBuffer agentBuffer)
  {
    // Complete the body of the receive exit here
  }
    // This method implements the security exit interface
  public ByteBuffer channelSecurityExit(
MQCXP channelExitParms,
                                      MQCD channelDefinition,
                                      ByteBuffer agentBuffer)
  {
    // Complete the body of the security exit here
  }
}
```

Her bir çıkış bir MQCXP nesnesi ve bir MQCD nesnesi değiştirgeleri olarak alır. Bu nesneler, yordamsal arabirimde tanımlanan MQCXP ve MQCD yapılarını gösterir.

Bir gönderme çıkışı çağrıldığında, agentBuffer parametresi, sunucu kuyruk yöneticisine gönderilmek üzere olan verileri içerir. agentBuffer.limit () ifadesi verilerin uzunluğunu sağladığından, uzunluk parametresi gerekli değildir. Çıkış gönderme işlevi, değeri sunucu kuyruk yöneticisine gönderilecek olan değeri olarak döndürülür. Ancak, gönderme çıkışı bir gönderme çıkışındaki son çıkış değilse, döndürülen veriler sırayla bir sonraki gönderme çıkışa aktarılır. Bir gönderme çıkışı, agentBuffer değiştirgesinde aldığı verilerin değiştirilmiş bir sürümünü döndürebilir ya da verileri değiştirmeden geri döndürebilir. Bu nedenle, olabilecek en basit çıkış gövdesi:

```
{ return agentBuffer; }
```

Bir alma çıkışı çağrıldığında, agentBuffer parametresi, sunucu kuyruk yöneticisinden alınan verileri içerir. Alma çıkışı, JMS için WebSphere MQ sınıflarına göre uygulamaya geçirilecek verilerin değeri olarak geri döner. Ancak, alma çıkışı bir alma işlemi sırasında son alma çıkışı değilse, döndürülen veriler sırayla bir sonraki alma çıkışa geçirilir.

Bir güvenlik çıkışı çağrıldığında, agentBuffer parametresi, bağlantının sunucu ucundaki güvenlik çıkışından bir güvenlik akımında alınan verileri içerir. Güvenlik çıkışı, bir güvenlik akışında sunucu güvenlik çıkışa gönderileceği veri değeri olarak döndürülür.

Kanal çıkışları, arka diziye sahip bir arabelleğle çağrılır. En iyi başarımı elde etmek için, çıkışta yedek diziye sahip bir arabellek döndürülmelidir.

Çağrıldığında, en çok 32 karakterlik kullanıcı verileri kanal çıkışa geçirilebilir. Çıkış, MQCXP nesnesinin getExitData () yöntemini çağırarak kullanıcı verilerine erişir. Çıkış, setExitData () yöntemini çağırarak kullanıcı verilerini değiştirebilse de, çıkış her çağrıldığında kullanıcı verileri yenilenir. Bu nedenle, kullanıcı verilerde yapılan değişiklikler kaybedilir. Ancak, çıkış, MQCXP nesnesinin çıkış kullanıcı alanını kullanarak bir çağrıdan diğerine veri geçirebilir. The exit accesses the exit user area by reference by calling the getExitUserArea() method.

Her çıkış sınıfının bir oluşturucusu olmalıdır. Oluşturucu, önceki örnekte gösterilen varsayılan oluşturucu olabilir ya da dizgi değiştirgesi olan bir oluşturucu olabilir. Oluşturucu, sınıftaki her çıkışa ilişkin çıkış sınıfının bir eşgörünümünü yaratmak için çağrılır. Bu nedenle, önceki örnekte, gönderme çıkışı için MyMQExits sınıfının bir örneği yaratılır; alma çıkışı için başka bir yönetim ortamı yaratılır ve güvenlik çıkışı için üçüncü bir yönetim ortamı yaratılır. Bir dizgi değiştirgesi içeren bir oluşturucu çağrıldığında, değiştirge, yönetim ortamının yaratılmakta olduğu kanal çıkışa geçirilen kullanıcı verilerini içerir. Bir çıkış sınıfının hem varsayılan oluşturucusu hem de tek bir değiştirge oluşturucusu varsa, tek parametre oluşturucusu öncelik kazanır.

Bağlantıyı kanal çıkışı içinden kapatmayın.

Veriler, bir bağlantının sunucu ucuna gönderildiğinde, herhangi bir kanal çıkışı çağrıldıktan *sonra* SSL şifrelemesi gerçekleştirilir. Benzer şekilde, bir bağlantının sunucu ucundan veri alındığında, herhangi bir kanal çıkışı çağrılmadan *önce* SSL şifre çözme işlemi gerçekleştirilir.

In versions of WebSphere MQ classes for JMS earlier than Version 7.0, channel exits were implemented using the interfaces MQSendExit, MQReceiveExit, and MQSecurityExit. Bu arabirimleri kullanmaya devam edebilirsiniz, ancak yeni arabirimler geliştirilmiş işlev ve başarım için tercih edilir.

Configuring IBM WebSphere MQ classes for JMS to use channel exits

Bir IBM WebSphere MQ classes for JMS uygulaması, uygulama bir kuyruk yöneticisine bağlandığında başlatılacak olan MQI kanalından kanal güvenliğini, gönderme ve alma çıkışlarını kullanabilir. Uygulama, Java, C ya da C + + içinde yazılan çıkışları kullanabilir. Uygulama, art arda çalıştırılan gönderme ya da alma çıkışlarını da kullanabilir.

Aşağıdaki özellikler, bir gönderme çıkışı ya da bir JMS bağlantısı tarafından kullanılan bir gönderme çıkışı sırası belirtmektedir:

- Bir MQConnectionFactory nesnesine ilişkin **SENDEXIT** özelliği.
- Gelen iletişim için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan etkinleştirme belirtimindeki **sendexit** özelliği,
- Çıkış iletişimi için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan bir ConnectionFactory nesnesindeki **sendexit** özelliği.

Özelliğin değeri, virgüllerle ayrılmış bir ya da daha çok öğeyi oluşturan bir dizedir. Her bir öğe, bir gönderme çıkışını aşağıdaki yollardan biriyle tanımlar:

- The name of a class that implements the `WMQSendExit` interface for a send exit written in Java.
- C ya da C + + içinde yazılmış bir gönderme çıkışı için *libraryName* (*entryPointAd*) biçiminde bir dizgi.

Benzer bir şekilde, aşağıdaki özellikler bağlantı tarafından kullanılan alma çıkışını ya da alma çıkışlarını belirtir:

- Bir MQConnectionFactory nesnesine ilişkin **RECEXIT** özelliği.
- Gelen iletişim için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan etkinleştirme belirtimindeki **receiveexit** özelliği,
- Çıkış iletişimi için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan bir ConnectionFactory nesnesindeki **receiveexit** özelliği.

Aşağıdaki özellikler, bağlantı tarafından kullanılan güvenlik çıkışını belirtir:

- Bir MQConnectionFactory nesnesine ilişkin **SECEXIT** özelliği.
- Gelen iletişim için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan etkinleştirme belirtimindeki **securityexit** özelliği,
- Çıkış iletişimi için IBM WebSphere MQ kaynak bağdaştırıcısı tarafından kullanılan bir ConnectionFactory nesnesindeki **securityexit** özelliği.

MQConnectionFactoriesiçin, IBM WebSphere MQ JMS yönetim aracını ya da IBM WebSphere MQ Explorerolanağını kullanarak **SENDEXIT**, **RECEXIT** ve **SECEXIT** özelliklerini ayarlayabilirsiniz. Alternatively, an application can set the properties by calling the `setSendExit()`, `setReceiveExit()`, and `setSecurityExit()` methods.

Kanal çıkışları, kendi sınıf yükleyicilerine göre yüklenir. Bir kanal çıkışı bulmak için sınıf yükleyici belirtilen siparişte aşağıdaki konumları arar.

1. **com.ibm.mq.cfg.ClientExitPath.JavaExitsClasspath** özelliği tarafından belirtilen sınıf yolu ya da IBM WebSphere MQ istemcisi yapılandırma dosyasının Channels kısmında **JavaExitsClassPath** özniteliği tarafından.
2. Java sistem özelliği **com.ibm.mq.exitClasspath**tarafından belirtilen sınıf yolu. Bu özelliğin artık kullanımdan kaldırıldığını unutmayın.
3. The IBM WebSphere MQ exits directory, as shown in Çizelge 130 sayfa 877. Sınıf yükleyici ilk olarak, Java arşiv (JAR) dosyalarında paketlenmeyen sınıf dosyaları için dizini arar. Kanal çıkışı bulunamazsa, sınıf yükleyici dizinde JAR dosyalarıyla arama yapar.

Çizelge 130. IBM WebSphere MQ çıkış dizini

Altyapı	**Dizin**
UNIX and Linux	`/var/mqm/exits` (32 bit kanal çıkışları) `/var/mqm/exits64` (64 bit kanal çıkışları)
Windows	*kuruluş_data_dzn*\çıkışlar Burada *kuruluş_veri_dzn* , kuruluş sırasında IBM WebSphere MQ veri kütükleri için seçtiğiniz dizindir. Varsayılan dizin `C:\Program Files\IBM\WebSphere MQ`dizinidir.

Not: Bir kanal çıkışı birden çok yerde varsa, IBM WebSphere MQ classes for JMS bulduğu ilk eşgörünümü yükler.

Sınıf yükleyicisinin üst öğesi, IBM WebSphere MQ classes for JMS' yi yüklemek için kullanılan sınıf yükleyicidir. Bu nedenle, üst sınıf yükleyicisinin önceki konumların hiçbirinde bulunamaması durumunda kanal çıkışı yüklemesi mümkündür. However, when you are using the IBM WebSphere MQ classes for JMS in an environment such as a JEE application server, you are not likely to be able to influence the choice of the parent class loader and so the class loader should be configured by setting the Java system property **com.ibm.mq.cfg.ClientExitPath.JavaExitsClasspath** on the application server.

Uygulamanız Java Security Manager etkinleştirilmiş olarak çalıştırılıyorsa, uygulamanın çalışmakta olduğu Java yürütme ortamı tarafından kullanılan ilke yapılandırma dosyası, bir kanal çıkış sınıfını yüklemek için gereken izinlere sahip olmalıdır. Bunun nasıl yapacağına ilişkin bilgi için Java Security Manager altındaki JMS uygulamaları için IBM MQ sınıflarının çalıştırılmasıbaşlıklı konuya bakın.

The `MQSendExit`, `MQReceiveExit`, and `MQSecurityExit` interfaces supplied with versions of IBM WebSphere MQ earlier than Version 7.0 are still supported. If you use channel exits that implement these interfaces, `com.ibm.mq.jar` must be present in the class path.

Kanal çıkışlarının C ' de nasıl yazılacağı ile ilgili bilgi için bkz. “İleti alışverişi kanallarına ilişkin kanal çıkışı programları” sayfa 378. Kanal çıkış programlarını, Çizelge 130 sayfa 877içinde gösterilen dizinde C ya da C++ dilinde yazılmış olarak saklamanız gerekir.

Uygulamanız bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, “JMS için IBM WebSphere MQ classes for JMS” sayfa 878başlıklı konuya bakın.

JMS için WebSphere MQ sınıflarını kullanırken kanal çıkışlarına geçirilecek kullanıcı verilerinin belirtilmesi

Çağrıldığında, en çok 32 karakterlik kullanıcı verileri kanal çıkışa geçirilebilir.

Bir MQConnectionFactory nesnesinin SENDEXITINIT özelliği, çağrıldığında her bir gönderme çıkışa geçirilecek kullanıcı verilerini belirtir. Özelliğin değeri, kullanıcı verilerinin virgülle ayrılmış bir ya da daha fazla öğesini içeren bir dizgidir. Dizgi içindeki kullanıcı verilerinin her bir öğesinin konumu, hangi gönderme çıkıştan çıkıldığında, çıkış gönderilerinde kullanıcı verilerinin aktarılacağı saptar. Örneğin, dizgideki kullanıcı verilerinin ilk öğesi, bir gönderme çıkışları sırasındaki ilk gönderme çıkışa geçirilir.

You can set the SENDEXITINIT property by using the WebSphere MQ JMS administration tool or WebSphere MQ Explorer. Diğer bir seçenek olarak, bir uygulama özelliği setSendExitInit() yöntemini çağırarak ayarlayabilir.

Benzer bir şekilde, bir ConnectionFactory nesnesinin RECEXITINIT özelliği, her alma çıkışa geçirilen kullanıcı verilerini belirtir ve SECEXITINIT özelliği, güvenlik çıkışa geçirilen kullanıcı verilerini belirtir. Bu özellikleri, WebSphere MQ JMS yönetim aracı ya da WebSphere MQ Explorer 'ı kullanarak ayarlayabilirsiniz. Alternatively, an application can set the properties by calling the setReceiveExitInit() and setSecurityExitInit() methods.

Kanal çıkışlarına geçirilen kullanıcı verilerini belirlerken aşağıdaki kurallara dikkat edin:

- Bir dizgideki kullanıcı verilerinin sayısı, bir sıradaki çıkışların sayısından fazlaysa, kullanıcı verilerinin fazla öğeleri yoksayılır.
- Bir dizgideki kullanıcı verilerinin sayısı, bir sıradaki çıkış sayısından azsa, kullanıcı verilerinin her belirlenmemiş öğesi boş bir dizgi olarak ayarlanır. Bir dizgi içinde art arda iki virgül ya da bir dizginin başlangıcındaki bir virgül, kullanıcı verilerinin belirlenmemiş bir öğesini de gösterir.

Bir uygulama, bir kuyruk yöneticisine bağlanmak için bir istemci kanal tanımlama çizelgesi (CCDT) kullanıyorsa, istemci bağlantı kanalı tanımlamasında belirtilen tüm kullanıcı verileri, çağrıldığında kanal çıkışlarına geçirilir. İstemci kanal tanımlama çizelgesini kullanma hakkında daha fazla bilgi için bkz. “JMS için IBM WebSphere MQ classes for JMS” sayfa 878.

JMS için IBM WebSphere MQ classes for JMS

JMS uygulamasına ilişkin bir IBM WebSphere MQ sınıfları, istemci kanal tanımlama çizelgesinde (CCDT) saklanan istemci bağlantısı kanal tanımlarını kullanabilir. CCDT ' yi kullanmak için bir ConnectionFactory nesnesi yapılandırıyorsunuz. Kullanımıyla ilgili bazı kısıtlamalar vardır.

Bir ConnectionFactory nesnesinin belirli özelliklerini ayarlayarak bir istemci bağlantı kanalı tanımı yaratmanın bir alternatifi olarak, bir IBM WebSphere MQ classes for JMS uygulaması, istemci kanal tanımlama çizelgesinde saklanan istemci bağlantı kanalı tanımlamalarını kullanabilir. Bu tanımlar, IBM WebSphere MQ Script komutları (MQSC) ya da IBM WebSphere MQ Programlanır Komut Biçimi (PCF) komutları tarafından yaratılır. Uygulama bir Connection nesnesi yarattığında, IBM WebSphere MQ classes for JMS , uygun bir istemci bağlantısı kanal tanımlaması için istemci kanal tanımlama çizelgesini arar ve bir MQI kanalı başlatmak için kanal tanımlamasını kullanır. İstemci kanal tanımlama tabloları ve nasıl oluşturulacağı hakkında daha fazla bilgi için bkz. İstemci kanal tanımlama tablosu.

Bir istemci kanal tanımlama çizelgesi kullanmak için, bir ConnectionFactory nesnesinin CCDTURL özelliğinin bir URL nesnesi olarak ayarlanması gerekir. URL nesnesi, istemci kanal tanımlama çizelgesini içeren dosyanın adını ve yerini tanımlayan ve dosyanın nasıl erişilebileceğini belirten bir URL adresi (uniform resource locator; URL) yerleştirir. CCDTURL özelliğini IBM WebSphere MQ JMS yönetim aracını kullanarak ayarlayabilir ya da bir uygulama, bir URL nesnesi yaratarak ve ConnectionFactory nesnesinin setCCDTURL() yöntemini çağırarak özelliği ayarlayabilirler.

Örneğin, ccdt1.tab dosyası bir istemci kanal tanımlama çizelgesi içeriyorsa ve uygulamanın çalıştığı sistemde saklandıysa, uygulama CCDTURL özelliğini aşağıdaki şekilde ayarlayabilir:

```
java.net.URL chanTab1 = new URL("file:///home/admdata/ccdt1.tab");
factory.setCCDTURL(chanTab1);
```

Başka bir örnek olarak, ccdt2.tab dosyasının bir istemci kanal tanımlama çizelgesi içerdiğini ve uygulamanın çalışmakta olduğu sistemden farklı bir sistemde saklandığı varsayıldığını varsayalım. Dosyaya FTP iletişim kuralı kullanılarak erişilebildiyse, uygulama CCDTURL özelliğini aşağıdaki şekilde ayarlayabilir:

```
java.net.URL chanTab2 = new URL("ftp://ftp.server/admdata/ccdt2.tab");
factory.setCCDTURL(chanTab2);
```

ConnectionFactory nesnesinin CCDTURL özelliğini ayarlamaya ek olarak, aynı nesnenin QVALER özelliği aşağıdaki değerlerden birine ayarlanmalıdır:

- Kuyruk yöneticisinin adı
- Bir yıldız işareti (*) ve ardından bir kuyruk yöneticisi grubu adı gelir.
- Yıldız işareti (*)
- Boş bir dizgi ya da tüm boşluk karakterlerini içeren bir dizgi

Bu değerler, Message Queue Interface (MQI) kullanan bir istemci uygulaması tarafından yayınlanan bir MQCONN çağrısındaki *QMgrName* parametresi için kullanılabilecek değerlerdir. Bu değerlerin anlamı hakkında daha fazla bilgi için bkz. MQCONN. You can set the QMANAGER property by using the WebSphere MQ JMS administration tool or IBM WebSphere MQ Explorer. Alternatively, an application can set the property by calling the setQueueManager() method of the ConnectionFactory object.

Bir uygulama ConnectionFactory nesnesinden bir bağlantı nesnesi yarattıysa, IBM WebSphere MQ classes for JMS , CCDTURL özelliği tarafından tanımlanan istemci kanalı tanımlama çizelgesine erişir, uygun bir istemci bağlantısı kanalı tanımlaması için çizelgeyi aramak için QMANMERER özelliğini kullanır ve sonra bir MQI kanalını kuyruk yöneticisine başlatmak için kanal tanımlamasını kullanır.

Uygulama createConnection() yöntemini çağırdığında, bir ConnectionFactory nesnesinin CCDTURL ve KANAL özelliklerinin her ikisi de belirlenemez. Her iki özellik de ayarlandıysa, yöntem bir kural dışı durum yayınlar. Değeri boş değerli, boş bir dizgi ya da tüm boş karakterleri içeren bir dizgi ise CCDTURL ya da CHANNEL özelliği ayarlanacak şekilde kabul edilir.

IBM WebSphere MQ classes for JMS , istemci kanal tanımlama çizelgesinde uygun bir istemci bağlantısı kanalı tanımlaması bulduğunda, bir MQI kanalını başlatmak için çizelgeden alınan bilgileri kullanır. ConnectionFactory nesnesinin kanalla ilgili özellikleri yok sayılır.

Güvenli Yuva Arabirimi Katmanı (SSL) kullanıyorsanız, özellikle aşağıdaki noktalara dikkat edin:

- Bir MQI kanalı, yalnızca istemci kanal tanımlama çizelgesinden alınan kanal tanımlaması IBM WebSphere MQ classes for JMStarafından desteklenen bir CipherSpec adını belirtiyorsa SSL kullanır.
- İstemci kanalı tanımlama çizelgesi, sertifika iptal listelerini (CRL ' ler) bulunduran LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) sunucularının konularıyla ilgili bilgileri de içerir. IBM WebSphere MQ classes for JMS , CRL ' leri tutan LDAP sunucularına erişmek için yalnızca bu bilgileri kullanır.
- Bir istemci kanal tanımlama çizelgesi, OCSP yanıtlayıcıya ait bir yeri de içerebilir. IBM WebSphere MQ classes for JMS , bir istemci kanal tanımlama çizelgesi dosyasında OCSP bilgilerini kullanamaz. Ancak, OCSP ' yi Using Online Certificate Protocol(Çevrimiçi Sertifika İletişim Kuralını Kullanma) bölümünde açıklandığı gibi yapılandırabilirsiniz.

İstemci kanal tanımlama çizelgesiyle SSL ' nin kullanılmasına ilişkin ek bilgi için Ek işlem istemcisinin SSL kanallarıyla kullanılmasıbaşlıklı konuya bakın.

Kanal çıkışlarını kullanıyorsanız aşağıdaki noktalara da dikkat edin:

- Bir MQI kanalı, istemci kanalı tanımlama çizelgesinden alınan kanal tanımlaması tarafından belirlenen kanal çıkışlarını ve ilişkili kullanıcı verilerini kullanır.

- İstemci kanal tanımlama çizelgesinden çıkarılan bir kanal tanımlaması, Java 'da yazılan kanal çıkışlarını belirleyebilir. Örneğin, bir istemci bağlantı kanalı tanımlaması yaratmak için DEFINE CHANNEL komutundaki SCYEXIT parametresinin, WMQSecurityExit arabirimini gerçekleştiren bir sınıfın adını belirtebileceği anlamına gelir. Benzer şekilde, SENDEXIT parametresi, WMQSendExit arabirimini gerçekleştiren sınıfın adını belirleyebilir ve RCVEXIT parametresi, WMQReceiveExit arabirimini gerçekleştiren sınıfın adını belirtebilir. Kanal çıkışısının Java 'da nasıl yazılabileceği hakkında daha fazla bilgi için bkz. “JMS için WebSphere MQ sınıfları için Java 'da kanal çıkışları yazılıyor” sayfa 875.

  Java dışında bir dilde yazılmış kanal çıkışlarının kullanılması da desteklenir. Başka bir dilde yazılmış kanal çıkışları için DEFINE CHANNEL komutunda SCYEXIT, SENDEXIT ve RCVEXIT parametrelerinin nasıl belirtileceği hakkında bilgi için DEFINE CHANNELbaşlıklı konuya bakın.

Otomatik JMS istemcisi yeniden bağlantısı

JMS istemcinizin konfigürasyonunu bir ağ, kuyruk yöneticisi ya da sunucu hatasının ardından otomatik olarak yeniden bağlanmanızı sağlar.

Bir bağlantı hatasını ya da bir kuyruk yöneticisini durdurduktan sonra istemci uygulamalarını yeniden bağlamak için bir yönetim isteğini ya da bağlantı hatasını takiben otomatik olarak yeniden bağlantı kurmak üzere istemci bağlantısı yapılandırmak için `MQConnectionFactory` sınıfının `CONNECTIONNAMELIST` ve `CLIENTRECONNECTOPTIONS` özelliklerini kullanın.

Bir connectionNamelistesindeki bağlantı adlarının tam listesi yalnızca, bağlantı adları listesini işleyebilen set/getconnectionNameadlı liste yönteminde erişilebilir. Ad listelerini işlemeyen, listedeki ilk ada erişmek için, get/setHostname gibi yöntemler.

Otomatik olarak yeniden bağlanabilir istemci bağlantıları, bağlantı kurulduktan sonra yalnızca yeniden bağlanabilir.

Bir uygulamanın otomatik olarak yeniden bağlandıktan sonra doğru şekilde çalışmaya devam edip etmeyeceğini, tasarımına bağlıdır. Yeniden bağlanabilir istemcilerin nasıl tasarlanması hakkında bilgi almak için ilgili konuları okuyun. Bazı var olan istemciler, otomatik yeniden bağlanma sonrasında değişiklik yapılmadan düzgün bir şekilde çalışabilir.

Otomatik istemci yeniden bağlanması, Java için WebSphere MQ sınıfları tarafından desteklenmez.

Başarısız olan bir kuyruk yöneticisine bağlı tüm istemcilerin eşzamanlı olarak yeniden bağlanmasını önlemek için, yeniden bağlanma girişimleri, kısmen sabit ve kısmen rasgele olan aralıklar tarafından geciktirilir.

Varsayılan olarak, yeniden bağlanma girişimleri şu aralıklarda gerçekleşir:

1. İlk deneme, bir saniyenin ilk gecikmesinden sonra, artı 250 milisaniyeye kadar olan rasgele bir öğelerden sonra yapılır.
2. İkinci deneme, ilk deneme başarısız olduktan sonra iki saniye, artı 500 milisaniyeye kadar rasgele bir aralıkla yapılır.
3. Üçüncü girişim dört saniye, ikincisi ise ikinci girişim başarısız olduktan sonra, bir saniyeye kadar olan rasgele bir aralıktan oluşur.
4. Dördüncü deneme sekiz saniye, artı üçüncü deneme başarısız olduktan sonra iki saniyeye kadar olan rasgele bir aralıkla yapılır.
5. Beşinci girişim 16 saniye, artı dördüncü girişim başarısız olduktan sonra, dört saniyeye kadar rasgele bir aralık haline getirilmektedir.
6. Altıncı girişim ve sonraki tüm denemeler 25 saniye, bir önceki girişim başarısız olduktan sonra, altı saniye ve 250 milisaniyeye kadar rasgele bir aralıkla yapılır.

Bu yeniden bağlanma işlemi, istemci kuyruk yöneticisine başarıyla yeniden bağlanıncaya kadar ya da yeniden bağlantı aralığı üst sınırı aralığı geçinceye kadar devam eder.

Kuyruk yöneticisinin kurtarılması için gereken süreyi daha doğru bir şekilde yansıtmak için varsayılan değerleri artırmanız gerekiyorsa ya da etkinleştirmek için beklemedeki kuyruk yöneticisi, MQCLIENT.INI kütüğünü kullanarak **ReconDelay** özniteliğini kullanın.

İlgili kavramlar

Otomatikleştirilmiş istemci yeniden bağlantısı

İlgili görevler

Yapılandırma dosyası kullanarak istemci yapılandırılması

IBM WebSphere MQ classes for JMSiçinde bir TCP/IP bağlantısının paylaşılması

Tek bir TCP/IP bağlantısını paylaşmak için bir MQI kanalının birden çok eşgörünümü yapılabilir.

Applications that are running inside the same Java runtime environment, and that use the IBM WebSphere MQ classes for JMS or the IBM WebSphere MQ resource adapter to connect to a queue manager by using the CLIENT transport, can be made to share the same channel instance.

Kanal örnekleri ve TCP/IP bağlantıları arasında bire bir ilişki vardır. Her kanal yönetim ortamı için bir TCP/IP bağlantısı yaratılır.

Bir kanal **SHARECNV** parametresiyle tanımlandıysa, 1değerinden büyük bir değere ayarlandıysa, bu sayıda etkileşim kanal yönetim ortamını paylaşabilir. Bir bağlantı üreticisini ya da etkinleştirme belirtimini bu işlevi kullanacak şekilde etkinleştirmek için, **SHARECONVALLOWED** özelliğini YES(EVET) olarak ayarlayın.

Bir JMS uygulaması tarafından yaratılan her JMS bağlantısı ve JMS oturumu, kuyruk yöneticisiyle kendi iletişimisini yaratır.

Bir etkinleştirme belirtimi başlatıldığında, JMS kaynak bağdaştırıcısına ilişkin IBM WebSphere MQ sınıfları, kullanılacak etkinleştirme belirtimine ilişkin kuyruk yöneticisiyle bir etkileşim başlatır. Sunucu oturumu havuzundaki etkinleştirme belirtimiyle ilişkilendirilmiş her sunucu oturumu, kuyruk yöneticisiyle de bir etkileşim başlatır.

SHARECNV özniteliği, bağlantı paylaşımına yönelik en iyi bir çalışma yaklaşımsıdır. Bu nedenle, IBM WebSphere MQ classes for JMSile 0 'dan büyük bir SHARECNV değeri kullanılırsa, yeni bir bağlantı isteğinin önceden kurulmuş bir bağlantıyı her zaman paylaşacağı garanti edilmez.

Kanal örneklerinin sayısını hesaplama

Bir uygulama tarafından yaratılan kanal somut örnekleri sayısı üst sınırını belirlemek için aşağıdaki formulae kullanın:

Etkinleştirme belirtimleri

Kanal örneklerinin sayısı = (<**maxPoolDepth**> + 1)/<**SHARECNV**>

Burada <**maxPoolDepth**>, **maxPoolDepth** özelliğinin değeri ve <**SHARECNV**>, etkinleştirme belirtimi tarafından kullanılan kanalda **SHARECNV** özelliğinin değeridir.

Diğer JMS uygulamaları

Kanal eşgörünümlerinin sayısı = (< JMS bağlantıları > + < JMS sessionları>)/<**SHARECNV**>

Burada < JMS bağlantıları >, uygulama tarafından yaratılan bağlantı sayısıdır; burada < JMS oturumları > uygulama tarafından yaratılan JMS oturumlarının sayısıdır ve <**SHARECNV**>, etkinleştirme belirtimi tarafından kullanılan kanaldaki **SHARECNV** özelliğinin değeridir.

Örnekler

The following examples show how to use the formulae to calculate the number of channel instances that are created on a queue manager by applications by using either the IBM WebSphere MQ classes for JMS or the IBM WebSphere MQ classes for JMS resource adapter.

JMS uygulaması örneği

JMS uygulaması bağlantısı, CLIENT iletmesini kullanarak bir kuyruk yöneticisine bağlanır ve JMS bağlantısı ve üç JMS oturumu yaratır. The channel that the application is using to connect to the queue manager has the **SHARECNV** property set to the value of 10. Uygulama çalışırken, uygulama ile kuyruk yöneticisi ile bir kanal yönetim ortamı arasında dört etkileşim vardır. Dört konuşmayla tüm kanal yönetim ortamı paylaşılıyor.

Etkinleştirme belirtimi örneği

Bir etkinleştirme belirtimi, CLIENT iletimi kullanılarak kuyruk yöneticisine bağlanır. Etkinleştirme belirtimi, **maxPoolDepth** özelliği 10 değerine ayarlanmış şekilde yapılandırılır. Etkinleştirme belirtiminin kullanmak üzere yapılandırıldığı kanalda **SHARECNV** özelliği 10 değerine ayarlanmış olmalıdır. Etkinleştirme belirtimi çalışırken 10 ileti eşzamanlı olarak işlenirken, etkinleştirme belirtimi ile kuyruk yöneticisi arasındaki etkileşim sayısı 11 (sunucu oturumları için 10 etkileşim ve etkinleştirme belirtimi için bir) olur. Etkinleştirme belirtimi tarafından kullanılan kanal eşgörünümlerinin sayısı 2 'dir.

Etkinleştirme belirtimi örneği

Bir etkinleştirme belirtimi, CLIENT iletimi kullanılarak kuyruk yöneticisine bağlanır. Etkinleştirme belirtimi, **maxPoolDepth** özelliği 5 olarak ayarlanmış şekilde yapılandırılır. Etkinleştirme belirtiminin kullanmak üzere yapılandırıldığı kanalda **SHARECNV** özelliği 0 olarak ayarlanmış olmalıdır. Etkinleştirme belirtimi çalıştırılırken ve koşutzamanlı 5 ileti işlenirken, etkinleştirme belirtimi ile kuyruk yöneticisi arasındaki etkileşim sayısı 6 (sunucu oturumları için beş etkileşim ve etkinleştirme belirtimi için bir) olur. Kanalın üzerindeki **SHARECNV** özelliği 0 değerine ayarlandığından, etkinleştirme belirtimi tarafından kullanılan kanal eşgörünümlerinin sayısı 0 'tır, her etkileşim kendi kanal yönetim ortamını kullanır.

JMS için WebSphere MQ sınıflarında istemci bağlantıları için bir kapı aralığı belirtme

Uygulamanızın bağlanabileceği bir kapı aralığı belirtmek için LOCALADDRESS özelliğini kullanın.

JMS uygulaması için bir WebSphere MQ sınıfları, istemci kipinde bir WebSphere MQ kuyruk yöneticisine bağlanmayı denediğinde, güvenlik duvarı yalnızca belirtilen kapılardan ya da bir kapı aralığından kaynaklanan bağlantılara izin verebilir. Bu durumda, uygulamanın bağlanabileceği bir kapı ya da kapı aralığı belirtmek için bir ConnectionFactory, QueueConnectionFactory ya da TopicConnectionFactory nesnesinin LOCALLEE özelliğini kullanabilirsiniz.

LOCALADDRESS özelliğini, WebSphere MQ JMS yönetim aracını kullanarak ya da bir JMS uygulamasında setLocalAddress () yöntemini çağırarak ayarlayabilirsiniz. Burada, bir uygulama içinden özelliğin ayarlanmasını gösteren bir örnek vardır:

```
mqConnectionFactory.setLocalAddress("192.0.2.0(2000,3000)");
```

Uygulama daha sonra bir kuyruk yöneticisine bağlandığında, uygulama 192.0.2.0(2000)- 192.0.2.0(3000) aralığındaki yerel bir IP adresine ve kapı numarasına bağlanır.

Birden çok ağ arabirimine sahip bir sistemde, bir bağlantı için hangi ağ arabiriminin kullanılması gerektiğini belirlemek için LOCALADDRESS özelliğini de kullanabilirsiniz.

Bir aracıya gerçek zamanlı bağlantı için, LOCALADDRESS özelliği yalnızca çoklu yayın kullanıldığında anlamlıdır. Bu durumda, bir bağlantı için hangi yerel ağ arabiriminin kullanılması gerektiğini belirtmek için özelliği kullanabilirsiniz, ancak özelliğin değeri bir kapı numarası ya da bir kapı numarası aralığı içermemelidir.

Kapı aralığını sınırladığınızda bağlantı hataları ortaya çıkabilir. Bir hata ortaya çıkarsa, WebSphere MQ neden kodu MQRC_Q_MGR_NOT_AVALABILIR ve şu iletiyi içeren yerleşik bir MQException ile bir JMSException yayınlanır:

```
LOCAL_ADDRESS_PROPERTY kısıtlaması nedeniyle yuva bağlantısı girişimi reddedildi
```

Belirtilen aralıktaki tüm kapılar kullanımdaysa ya da belirtilen IP adresi, anasistem adı ya da kapı numarası geçerli değilse (örneğin, negatif bir kapı numarası) bir hata oluşabilir.

JMS için WebSphere MQ sınıfları, bir uygulama için gerekenlerden farklı bağlantılar yaratabileceğinden, her zaman bir kapı aralığı belirtmeyi düşünebilirsiniz. Genel olarak, bir uygulama tarafından oluşturulan her oturum için bir kapı gerekir ve JMS için WebSphere MQ sınıfları üç ya da dört ek kapı gerektirebilir. Bir bağlantı hatası ortaya çıkarsa, kapı aralığını artırın.

JMS için WebSphere MQ sınıflarında varsayılan olarak kullanılan bağlantı havuzlama, kapıların yeniden kullanılabileceği hız üzerinde bir etkiye sahip olabilir. Sonuç olarak, bağlantı noktaları serbest bırakılırken bir bağlantı hatası ortaya çıkabilir.

JMS için WebSphere MQ sınıflarında kanal sıkıştırması

JMS uygulaması için bir WebSphere MQ sınıfları, bir ileti üstbilgisini ya da verilerini sıkıştırmak için WebSphere MQ olanağını kullanabilir.

Bir WebSphere MQ kanalında akan verilerin sıkıştırılmasına ilişkin sıkıştırma, kanalın başarımını artırabilir ve ağ trafiğini azaltabilir. WebSphere MQile sağlanan işlevi kullanarak, ileti kanallarında ve MQI kanallarında akan verileri sıkıştırabilirsiniz. Her iki tipteki bir kanalda, üstbilgi verilerini ve ileti verilerini birbirinden bağımsız olarak sıkıştırabilirsiniz. Varsayılan olarak, bir kanalda veri sıkıştırılmadı.

A WebSphere MQ classes for JMS application specifies the techniques that can be used for compressing header or message data on a connection by creating a java.util.Collection object. Her sıkıştırma tekniği, kaynak grubundaki bir Tamsayı nesnesidir ve uygulamanın kaynak grubuna sıkıştırma tekniklerini eklediği sıra, uygulama bağlantıyı yarattığında sıkıştırma tekniklerinin kuyruk yöneticisiyle anlaşmalı olduğu sıradır. The application can then pass the collection to a ConnectionFactory object by calling the setHdrCompList() method, for header data, or the setMsgCompList() method, for message data. Uygulama hazır olduğunda, bağlantı yaratabilir.

Aşağıdaki kod parçaları açıklanan yaklaşımı gösterir. İlk kod parçası, üstbilgi veri sıkıştırmasının nasıl gerçekleştirileceğini gösterir:

```
Collection headerComp = new Vector();
headerComp.add(new Integer(WMQConstants.WMQ_COMPHDR_SYSTEM));
.
.
.
((MQConnectionFactory) cf).setHdrCompList(headerComp);
.
.
.
connection = cf.createConnection();
```

İkinci kod parçası, ileti veri sıkıştırmasının nasıl gerçekleştirileceğini gösterir:

```
Collection msgComp = new Vector();
msgComp.add(new Integer(WMQConstants.WMQ_COMPMSG_RLE));
msgComp.add(new Integer(WMQConstants.WMQ_COMPMSG_ZLIBHIGH));
.
.
.
((MQConnectionFactory) cf).setMsgCompList(msgComp);
.
.
.
connection = cf.createConnection();
```

İkinci örnekte, sıkıştırma teknikleri, bağlantı yaratıldığında sırasıyla RLE, daha sonra ZLIBLIHH olarak kararlaştırılır. Seçilen sıkıştırma tekniği, Connection nesnesinin geçerlik süresi boyunca değiştirilemez. Bir bağlantıda sıkıştırmayı kullanmak için, Connection nesnesi yaratılmadan önce setHdrCompList() ve setMsgCompList() yöntemlerinin çağrılması gerekir.

JMS için IBM WebSphere MQ sınıflarında ileti zamanuyumsuz olarak ileti konması

Olağan durumda, bir uygulama bir hedefe ileti gönderdiğinde, uygulamanın, kuyruk yöneticisinin isteği işlediğini doğrulamasını beklemesi gerekir. İletileri zamanuyumsuz olarak koymak yerine, bazı durumlarda ileti alışverişi başarımını artırabilirsiniz. Bir uygulama bir iletiyi zamanuyumsuz olarak yerleştirdiğinde, kuyruk yöneticisi her çağrımın başarısını ya da hatasını döndürmez, ancak bunun yerine düzenli aralıklarla hata denetimi yapabilirsiniz.

Bir hedefin, kuyruk yöneticisinin iletiyi güvenli bir şekilde alıp almadığı belirlenmeden, uygulamaya denetim işlevinin geri döndürülüp döndürülmeyeceği, aşağıdaki özelliklere bağlıdır:

- JMS Hedef Özelliği AçıKLAMA (kısa ad-PAALD).

  JMS uygulamalarının, JMS Hedefinin temsil ettiği temel kuyruk ya da konu bu seçeneğe izin veriyorsa, JMS uygulamalarının iletileri zamanuyumsuz olarak gönderip koyamayacağını denetler.
- IBM WebSphere MQ kuyruğu ya da konu özelliği DEFPRESP (Varsayılan put yanıt tipi).

DEFPRESP, kuyruğa ileti koyan uygulamaların ya da konuya ileti yayınlayıp yayınlamayacağını belirtir; zamanuyumsuz koyma işlevlerinden birini kullanabilir.

Aşağıdaki çizelge, PUTASINCALLED ve DEFPRESP özelliklerinin olası değerlerini ve zamanuyumsuz koyma işlevinin geçerli kılınmasını gerektiren değerleri göstermektedir:

Çizelge 131. İletilerin zamanuyumsuz olarak konulması durumunda, PUTASINCALLI ve DEFPRESP özellikleri saptanıyor.

WebSphere MQ kuyruk özelliği	**PUTASYNCALDIC = NO.**	**PUTASINCALLI = YES (EVET)**	**PUTASYNCALLID = AS_DEST ya da AS_Q_DEF ya da AS_T_DEF**
DEFPRESP=SYNC	Zamanuyumsuz put işlevi etkin değil	Zamanuyumsuz koyma işlevi etkinleştirildi	Zamanuyumsuz put işlevi etkin değil
DEFPRESP=ASYNC	Zamanuyumsuz put işlevi etkin değil	Zamanuyumsuz koyma işlevi etkinleştirildi	Zamanuyumsuz koyma işlevi etkinleştirildi

For messages sent in a transacted session, the application ultimately determines whether the queue manager has received the messages safely when it calls `commit()`.

Bir uygulama, hareket eden bir oturum içinde kalıcı iletiler gönderirse ve bir ya da daha fazla ileti güvenli bir şekilde alınmazsa, işlem kesinleştirilemez ve kural dışı durum üretir. Ancak, bir uygulama hareket edilen bir oturum içinde kalıcı olmayan iletiler gönderirse ve bir ya da daha fazla ileti güvenli bir şekilde alınmazsa, işlem başarıyla kesinleştirilir. Uygulama, kalıcı olmayan iletilerin güvenli bir şekilde ulaşmadığına ilişkin herhangi bir geri bildirim almaz.

İşlem dışı bir oturumda gönderilen kalıcı olmayan iletiler için, *ConnectionFactory* nesnesinin SENDCHECKCOUNT özelliği, kuyruk yöneticisinin iletileri güvenli bir şekilde aldığı JMS denetimlerine ilişkin IBM WebSphere MQ sınıflarından önce kaç iletinin gönderileceğini belirtir.

Bir denetim, bir ya da daha çok iletinin güvenli bir şekilde alınmadığını ve uygulamanın bağlantı ile bir kural dışı durum dinleyicisi kaydettiyse, JMS için IBM WebSphere MQ sınıfları, kural dışı durum dinleyicisinin `onException()` yöntemini uygulamaya bir JMS kural dışı durumunu geçirmesi için çağırmaktadır.

JMS kural dışı durumu JMSWMQ0028 hata koduna sahip ve bu kod şu iletiyi görüntüler:

```
At least one asynchronous put message failed or gave a warning.
```

JMS kural dışı durumunun ayrıca daha fazla ayrıntı sağlayan bağlantılı bir kural dışı durumu da vardır. SENDCHECKCOUNT özelliğinin varsayılan değeri sıfır olup bu, bu tür denetlerin yapılmadığı anlamına gelir.

Bu eniyileme, istemci kipinde bir kuyruk yöneticisine bağlanan ve bir dizi iletiyi hızlı bir şekilde gönderip göndermesi gereken bir uygulamaya en çok yarar sağlar, ancak gönderilen her ileti için kuyruk yöneticisinden hemen geribildirim gerektirmez. Ancak, bir uygulama bağ tanımları kipindeki bir kuyruk yöneticisine bağlansa da, bu eniyilemeyi yine de kullanabilir, ancak beklenen performans avantajı da bu kadar büyük değildir.

JMS için WebSphere MQ sınıflarıyla birlikte okuma seçeneğini kullanma

WebSphere MQ tarafından sağlanan okuma tamamlama işlevi, bir işlemin dışında alınan kalıcı olmayan iletilerin, bir uygulama tarafından istekte bulunmadan önce IBM WebSphere MQ classes for JMS ' ye gönderilmesini sağlar. IBM WebSphere MQ classes for JMS , iletileri bir iç arabelleğiyle saklar ve uygulama bunları sorduğunda, iletileri uygulamaya geçirir.

Bir hareketin dışındaki bir hedeften gelen iletileri almak için `MessageConsumers` ya da `MessageListeners` kullananIBM WebSphere MQ classes for JMS uygulamaları, ileriye okuma işlevini kullanabilir. İleriyi okumanın kullanılması, bu nesneleri kullanan uygulamaların ileti aldıklarında daha iyi bir performansa neden olacak şekilde yararlanmasını sağlar.

Whether an application that uses `MessageConsumers` or `MessageListeners` can use read ahead depends upon the following properties:

- READAHEADINE (kısa ad-RAALD) JMS Hedef Özelliği. READAHEADIZIN, JMS uygulamalarının bir hareketin dışında kalıcı olmayan iletileri alırken ya da JMS Hedefinin gösterdiği temel kuyruk ya da

konu dışında, okuma öncesinde okuma kullanıp kullanamayacağını denetler; bu seçenek bu seçeneği sağlar.

- IBM WebSphere MQ kuyruğu ya da konu özelliği DEFREADA (Varsayılan okuma değeri). DEFREADA, bir hareketin dışındaki kalıcı olmayan iletilerin alılıp alışmadığını belirten uygulamaların önden okuma kullanabileceğini belirtir.

Aşağıdaki çizelge, READAHEADIZIN ve DEFREADA özelliklerine ilişkin olası değerleri ve öndeki okuma işlevinin etkinleştirilmesini gerektiren değerleri göstermektedir:

Çizelge 132. READAHEADIZIN ve DEFREADA özellikleri, bir hareket dışında kalıcı olmayan iletiler alınırken ya da göz atılırken okuma öncesinde kullanılıp kullanılmadığı saptanıyor.

WebSphere MQ hedef özelliği	**READAHEADALLOWED = YES**	**READAHEADALLOWED = NO**	**AS_DEST ya da AS_Q_DEF ya da AS_T_DEF**
WebSphere MQ kuyruk özelliği			
DEFREADA = NO	Okuma tamamlama işlevi etkin	Okuma öncesinde okuma işlevi etkinleştirilmedi	Okuma öncesinde okuma işlevi etkinleştirilmedi
DEFREADA = EVET	Okuma tamamlama işlevi etkin	Okuma öncesinde okuma işlevi etkinleştirilmedi	Okuma tamamlama işlevi etkin
DEFREACA = DEVRE Dışı	Okuma öncesinde okuma işlevi etkinleştirilmedi	Okuma öncesinde okuma işlevi etkinleştirilmedi	Okuma öncesinde okuma işlevi etkinleştirilmedi

İleriye okuma işlevi etkinleştirilirse, uygulama tarafından bir `MessageConsumer` ya da `MessageListener` yaratıldığında, IBM WebSphere MQ classes for JMS , `MessageConsumer` ya da `MessageListener` ' in izlediği hedef için bir iç arabellek yaratır. Her `MessageConsumer` ya da `MessageListener`için bir iç arabellek var. Uygulama, aşağıdaki yöntemlerden birini çağırdığında, kuyruk yöneticisi kalıcı olmayan iletileri IBM WebSphere MQ classes for JMS ' e göndermeye başlar:

- `MessageConsumer.receive()`
- `MessageConsumer.receive(long timeout)`
- `MessageConsumer.receiveNoWait()`
- `Session.setMessageListener(MessageListener listener)`

The IBM WebSphere MQ classes for JMS automatically returns the first message back to the application, by the method call that the application has made. Kalıcı olmayan diğer iletiler, hedef için yaratılmış iç arabelleğde IBM WebSphere MQ classes for JMS tarafından saklanır. Uygulama bir sonraki iletiyi işlemeye istediğinde, IBM WebSphere MQ classes for JMS iç arabelleğindeki bir sonraki iletiyi döndürür.

The IBM WebSphere MQ classes for JMS requests more non-persistent messages from the queue manager when the internal buffer is empty.

The internal buffer that is used by the IBM WebSphere MQ classes for JMS is deleted when an application closes a `MessageConsumer`, or the JMS Session that a `MessageListener` is associated with.

`MessageConsumer`siçin, iç arabelleğindeki işlenmemiş iletiler kaybedilir.

`MessageListener`skullanıldığında, iç arabelleğindeki iletilere ne olur, READAHEADCLOSEPOLICY (kısa ad-RACP) JMS hedef özelliğine bağlıdır. Özelliğin varsayılan değeri DELIVER_ALL olur; bu, `MessageListener` oluşturmak için kullanılan JMS oturumunun, iç arabelleğindeki tüm iletiler uygulamaya teslim edilinceye kadar kapatılmamasını sağlar. Özellik DELIVER_CURRENT olarak ayarlanırsa, yürürlükteki ileti uygulama tarafından işlendikten sonra JMS oturumu kapatılır ve iç arabelleğindeki geri kalan tüm iletiler atılır.

JMS için WebSphere MQ sınıflarında yayınları alıkoyma

JMS istemcisi için bir WebSphere MQ sınıfları saklanan yayınlarını kullanacak şekilde yapılandırılabilir.

Bir yayınlayıcı, konuya ilgi duyan gelecekteki abonelere gönderilebilmesi için bir yayının kopyasının saklaması gerektiğini belirtebilir. Bu, JMS_IBM_RETAIN tamsayı özelliğini değer 1 değerine ayarlayarak JMS için WebSphere MQ sınıflarında gerçekleştirilir. Sabit değerler, com.ibm.msg.client.jms.JmsConstants arabiriminde bu değerler için tanımlanmış olmalıdır. Örneğin, alıkonan bir yayın olarak ayarlamak için *İlt*adlı bir ileti oluşturdıysanız şu kodu kullanın:

```
// set as a retained publication
msg.setIntProperty(JmsConstants.JMS_IBM_RETAIN, JmsConstants.RETAIN_PUBLICATION);
```

Şimdi iletiyi normal olarak gönderebilirsiniz. JMS_IBM_RETAIN, alınan bir iletide de sorgulanabilir. Bu nedenle, alınan iletinin alıkonan bir yayın olup olmadığını sorgulamak mümkündür.

JMS için WebSphere MQ sınıflarında XA desteği

JMS, bağ tanımlarında ve istemci kiplerindeki XA uyumlu hareketleri desteklenen bir hareket yöneticisiyle destekler.

Bir uygulama sunucusu ortamında XA işlevselliğine gereksinim duyarsanız, uygulamanızı uygun şekilde yapılandırmanız gerekir. Dağıtılmış hareketleri kullanmak için uygulamaların nasıl yapılandırılacasına ilişkin bilgi edinmek için uygulama sunucunuzun kendi belgelerine bakın.

WebSphere Event Broker ya da WebSphere Message Broker aracısıyla gerçek zamanlı bir bağlantı kullanılması

JMS uygulaması için bir WebSphere MQ sınıfları, yayınlama/abone olma ileti alışverişi için WebSphere Event Broker ya da WebSphere Message Broker aracısına gerçek zamanlı bir bağlantı kullanabilir. JMS için hem aracı hem de WebSphere MQ sınıfları, gerçek zamanlı bir bağlantı etkinleştirmek için yapılandırılmalıdır.

When an application uses a real-time connection to a broker of WebSphere Event Broker or WebSphere Message Broker, the application and the broker exchange messages using WebSphere MQ Real-Time Transport. Depending on the configuration, messages can also be delivered to the application using WebSphere MQ Multicast Transport.

For information about how an application can connect to a WebSphere MQ queue manager and use WebSphere MQ Enterprise Transport to exchange messages with a broker of WebSphere Event Broker or WebSphere Message Broker, see the documentation for previous releases of WebSphere MQ classes for JMS. WebSphere MQ Enterprise Transport olanağını kullanabilmek için, bir uygulamanın WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipinde çalışan bir bağlantı üreticisi kullanarak bir kuyruk yöneticisine bağlanması gerektiğini unutmayın.

Gerçek zamanlı bağlantı için WebSphere Event Broker ya da WebSphere Message Broker aracısının yapılandırılması

JMS uygulamasının bir WebSphere MQ sınıflarında, WebSphere Event Broker ya da WebSphere Message Broker aracısıyla gerçek zamanlı bağlantı kullanması için, aracının dinlediği TCP/IP kapısındaki iletileri okumak ve iletileri yayınlamak için ileti akışını yaratarak ve konuşlandırarak aracıyı yapılandırmalısınız. Gereksinimlerinize bağlı olarak, aracıyı ek yöntemlerle yapılandırmanız gerekebilir.

Aracıyı yapılandırmak için, aşağıdaki ileti akışlarından birini yaratmanız ve konuşlandırmanız gerekir:

- Real-timeOptimizedAkışı ileti işleme düğümünü içeren bir ileti akışı
- Gerçek-timeInput ileti işleme düğümü ve bir Yayın iletisi işleme düğümü içeren ileti akışı

Gerçek zamanlı bağlantılar için kullanılan TCP/IP bağlantı noktasını dinlemek için Real-timeOptimizedFlow ya da Real-timeInput düğümünü yapılandırmalısınız. Varsayılan olarak, gerçek zamanlı bağlantılar için kapı numarası 1506 'tır.

Aşağıdaki gereksinimlerden herhangi birine sahipseniz, aracıyı da yapılandırmanız gerekir:

- Uygulamanın SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) kimlik doğrulamasını kullanarak aracıya bağlanmasını istiyorsanız
- Uygulamanın HTTP tünellemesi kullanarak aracıya bağlanmasını istiyorsanız
- İletilerin çoklu yayın kullanan bir ileti tüketticisine teslim edilmesini istiyorsanız

Aracının nasıl yapılandırılacağı hakkında bilgi için *WebSphere Event Broker ürün belgeleri* ya da *WebSphere Message Broker ürün belgeleri*başlıklı konuya bakın.

Bir WebSphere Event Broker ya da WebSphere Message Broker aracısına gerçek zamanlı bağlantı için WebSphere MQ sınıflarının JMS için yapılandırılması

JMS uygulamasının bir WebSphere MQ sınıflarında, WebSphere Event Broker ya da WebSphere Message Broker aracısıyla gerçek zamanlı bir bağlantı kullanması için, bağlantı üreticisinin bazı özellikleri ayarlanarak JMS için WebSphere MQ sınıfları yapılandırılmalıdır. Gereksinimlerinize bağlı olarak, JMS için WebSphere MQ sınıflarının ek yollarla yapılandırılması gerekebilir.

JMS için WebSphere MQ sınıflarını yapılandırmak için, bağlantı üreticisinin aşağıdaki özellikleri ayarlanmalıdır:

- TRANSPORT özelliği, DIRECT olarak ayarlanmalıdır.

  Ancak, bir uygulamanın HTTP tünelleme kullanılarak bağlanması için, TRANSPORT özelliği DIRECTTHTTP olarak ayarlanmalıdır. Bkz. “HTTP tünellemesi kullanılıyor” sayfa 888.
- HOSTNAME özelliği, aracının çalışmakta olduğu sistemin anasistem adı ya da IP adresine ayarlanmalıdır.
- PORT (Kapı) özelliği, aracının gerçek zamanlı bağlantıları dinlediği kapının numarasına ayarlanmalıdır.

An application can set these properties dynamically at run time by using the IBM JMS extensions or the WebSphere MQ JMS extensions. Diğer bir seçenek olarak, bağlantı üreticisi yönetilen bir nesnelse, bir denetimci WebSphere MQ JMS yönetim aracı ya da WebSphere MQ Explorer olanağını kullanarak bu özellikleri ayarlayabilir.

Özellikler ve uygulamaların değerlerini ayarlamak için kullandığı yöntemler hakkında bilgi için IBM WebSphere MQ classes for JMS nesnelerinin özellikleribaşlıklı konuya bakın. WebSphere MQ JMS yönetim aracını nasıl kullanabilmeye ilişkin bilgi için bkz. “WebSphere MQ JMS yönetim aracının kullanılması” sayfa 896. WebSphere MQ Explorer olanağının nasıl kullanılacağı hakkında bilgi için WebSphere MQ Explorer ile sağlanan yardım konusuna bakın.

Aşağıdaki gereksinimlerin herhangi biri varsa, JMS için WebSphere MQ sınıfları ek yapılandırma gerektirir:

- Bir uygulamanın SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) kimlik doğrulamasını kullanarak aracıya bağlanmasını istiyorsanız
- Bir uygulamanın, HTTP tünellemesi kullanılarak aracıya bağlanmasını istiyorsanız
- Bir uygulamanın yetkili sunucu aracılığıyla aracıya bağlanmasını istiyorsanız
- İletilerin çoklu yayın kullanan bir ileti tüketticisine teslim edilmesini istiyorsanız

Aşağıdaki kısımlarda, bu gereksinimlerin her biri için JMS için WebSphere MQ sınıflarının nasıl yapılandırılacağı açıklanmaktadır.

SSL (Güvenli Yuva Arabirimi Katmanı) kimlik doğrulaması kullanılıyor

SSL kimlik doğrulaması, bir aracıya gerçek zamanlı bir bağlantıda kullanılabilir. Bu bağlantı tipi için yalnızca kimlik doğrulama desteklenir. Uygulama ile aracı arasında akan ileti verilerini şifrelemek ve şifrelerini çözmek ya da verilerin kurcalamayı saptamak için SSL ' yi kullanamazsınız.

Bu durum ile bir uygulama istemci kipinde bir kuyruk yöneticisine bağlandığında bu durum arasındaki farkı göz önünde bulundurun. In the latter case, you can use the WebSphere MQ SSL support to encrypt and decrypt the message data that flows between the application and the queue manager and to detect tampering of the data, as well as to provide authentication.

İleti verilerinin bir aracıya gerçek zamanlı bir bağlantıyla korunmasını istiyorsanız, aracı tarafından sağlanan işlevi kullanabilirsiniz. Korumak istediğiniz iletilerle her konuya bir koruma kalitesi (QoP) değeri

atayabilirsiniz. Bu nedenle, her konu için farklı bir ileti koruması düzeyi seçebilirsiniz. Aracı tarafından sağlanan ileti koruması hakkında daha fazla bilgi için, *WebSphere Event Broker ürün belgelerine* ya da *WebSphere Message Broker ürün belgelerine*bakın.

Bir aracıya gerçek zamanlı bir bağlantıda SSL kimlik doğrulamasını kullanmak için, bağlantı üreticisinin DIRECTEPUTH özelliğinin CERTIFICATE olarak ayarlanması gerekir.

Karşılıklı kimlik doğrulaması için SSL kullanmak istiyorsanız, aracının Kimlik Doğrulama İletişim Kuralı Tipi özelliği, simetrik SSL için R seçeneğini belirtmelidir. SSL ' yi yalnızca aracıyı doğrulamak için kullanmak istiyorsanız, aracının Authentication Protocol Type (Kimlik Doğrulama Protokolü) özelliği asimetrik SSL seçeneğini belirtmelidir. Ancak, bu durumda uygulama, aşağıdaki örnekte olduğu gibi, parametre olarak bir kullanıcı kimliği ve parolayla createConnection() çağrılarak aracıya bağlanmalıdır:

```
factory.createConnection("user1", "user1pw");
```

Daha sonra aracı, uygulamayı doğrulamak için SSL yerine kullanıcı kimliğini ve parolayı kullanır. Aracıyı SSL kimlik doğrulaması için nasıl yapılandırabilmeye ilişkin ek bilgi için *WebSphere Event Broker ürün belgeleri* ya da *WebSphere Message Broker ürün belgeleri*başlıklı konuya bakın.

Notlar:

1. DIRECTAUTH özelliğinin değeri, SSL kimlik doğrulamasının SSLCIPHERSUIT özelliğinin değeri değil, bir aracıya gerçek zamanlı bir bağlantıda kullanılıp kullanılmadığını belirler.
2. SSL kimlik doğrulaması bir aracıya gerçek zamanlı bir bağlantıda kullanıldığında, bir uygulama istemci kipinde bir kuyruk yöneticisine bağlandığında gerçekleştirilenler ile aynı denetimlerini gerçekleştirmek için SSLPEERNAME ve SSLCRL özellikleri kullanılır.
3. JMS için WebSphere MQ sınıfları, aşağıdaki durumlarda SSL desteğini sağlamak için aynı Java Secure Socket Extension (JSSE) anahtar deposunu ve güvenilirlik deposu yapılandırmasını kullanabilir:
   - Bir uygulama bir aracıya gerçek zamanlı bağlantı kullandığında
   - Bir uygulama istemci kipindeki bir kuyruk yöneticisine bağlandığında

HTTP tünellemesi kullanılıyor

JMS uygulaması için bir WebSphere MQ sınıfları HTTP tünellemesi kullanılarak bir aracıya bağlanabilir. Bu, uygulamanın bir web sitesine bağlanmasına rağmen HTTP protokolünü kullanarak aracıya bağlandığı anlamına gelir.

Bir aracıya gerçek zamanlı bir bağlantıda HTTP tünelini kullanmak için, bağlantı üreticisinin TRANSPORT özelliği DIRECTTHTTP olarak ayarlanmalıdır.

HTTP tünellemesi, SSL kimlik doğrulamasıyla birlikte kullanılamaz, yetkili sunucu aracılığıyla bağlantı kurma ya da çoklu yayın kullanan iletiler sağlanamaz. HTTP protokolünün desteklenen sürümü: 1.0. HTTP sürüm 1.1 desteklenmiyor.

Yetkili sunucu aracılığıyla bağlanma

JMS uygulaması için bir WebSphere MQ sınıfları, yetkili sunucu aracılığıyla bağlantı kurarak bir aracıya gerçek zamanlı bağlantı kullanabilir. WebSphere MQ classes for JMS, doğrudan yetkili sunucuya bağlanır ve yetkili sunucudan aracıya bağlantı isteğini iletmesini istemek için RFC 2817 'de tanımlanan Internet iletişim kuralını kullanır.

Yetkili sunucu aracılığıyla bir aracıya bağlanmak için, bağlantı üreticisinin aşağıdaki özellikleri ayarlanmalıdır:

- PROXYHOSTNAME özelliği, yetkili sunucunun çalışmakta olduğu sistemin anasistem adı ya da IP adresi olarak ayarlanmalıdır.
- PROXYPT özelliği, yetkili sunucunun dinlemede olduğu kapının numarasına ayarlanmalıdır.

PROXYHOSTNAME özelliği ayarlanmadıysa ya da boş dizgi olarak ayarlandıysa, JMS için WebSphere MQ sınıfları yalnızca HOSTNAME ve PORT özelliklerini kullanarak doğrudan aracıya bağlanmayı dener ve bir yetkili sunucu aracılığıyla bağlanmayı denemez.

Çoklu yayını kullanarak ileti teslim etme

Bir aracıya gerçek zamanlı bağlantı kullanarak, iletiler çoklu yayın kullanılarak ileti tüketicisine teslim edilebilir.

Çoklu yayını etkinleştirmek için, Konu nesnesinin MULTICAST özelliği gereken çok noktaya gönderim seçeneğine ayarlanmalıdır. Diğer bir seçenek olarak, Konu nesnesinin MULTICAST özelliği ASCF olarak ayarlandıysa, bağlantı üreticisinin MULTICAST özelliği gereken çok noktaya gönderim seçeneğine ayarlanmış olmalıdır.

JMS için WebSphere MQ sınıfları hem Paket Aktarım Katmanı 'nı (PTL) hem de PGM (PGM) çoklu yayın iletişim kurallarını destekler ve PGM protokolünün, PGM/IP ve PGM UDP tarafından kapsüllenmiş her iki somutlama için de destek içerir. Ancak, PGM/IP desteği yalnızca aşağıdaki altyapılarda kullanılabilir:

- AIX (yalnızca 32 bit)
- Linux (x86 platformu)
- Linux (zSeries platform, 32-bit only)
- Solaris SPARC (yalnızca 32 bit)
- Windows (yalnızca 32 bit)
- z/OS

JMS Application Server Facilitis için WebSphere MQ sınıfları

Bu konuda, JMS için WebSphere MQ sınıflarının Oturum sınıfındaki ConnectionConsumer sınıfını ve gelişmiş işlevselliğini nasıl gerçekleştirdiğini ele alır. Ayrıca, bir sunucu oturum havuzunun işlevi de özetlenir.

WebSphere MQ classes for JMS supports the Application Server Facilities (ASF) that are specified in the *Java İleti Hizmeti Belirtimi, Sürüm 1.1* (see Sun's Java website at https://java.sun.com). Bu belirtim, bu programlama modeli içindeki üç rolü tanımlar:

- **JMS sağlayıcısı** , ConnectionConsumer ve gelişmiş Oturum işlevselliği sağlar.
- **Uygulama sunucusu** , ServerSessionhavuzu ve ServerSession işlevlerini sağlar.
- **İstemci uygulaması** , JMS sağlayıcısının ve uygulama sunucusu kaynağının işlevselliğini kullanır.

Bir uygulama bir aracıya gerçek zamanlı bağlantı kullanıyorsa, bu konudaki bilgiler geçerli değildir.

JMS ConnectionConsumer

ConnectionConsumer arabirimi, iletileri bir iş parçacığı havuzuna eşzamanlı olarak teslim etmek için yüksek performanslı bir yöntem sağlar.

JMS belirtimi, bir uygulama sunucusunun `ConnectionConsumer` arabirimini kullanarak JMS uygulaması ile yakından bütünleşmesini sağlar. Bu özellik iletilerin eşzamanlı işlenmesini sağlar. Tipik olarak, bir uygulama sunucusu bir iş parçacığı havuzu yaratır ve JMS somutlaması bu iş parçacıklarının kullanımına sunulan iletileri yapar. JMS tanıyan bir uygulama sunucusu (örneğin, WebSphere Application Server gibi), ileti odaklı Bean 'ler gibi üst düzey ileti sistemi işlevselliği sağlamak için bu özelliği kullanabilir.

Normal uygulamalar ConnectionConsumer' ı kullanmaz, ancak uzman JMS istemcileri bunu kullanabilir. Bu tür istemciler için, ConnectionConsumer , iletileri bir iş parçacığı havuzuna eşzamanlı olarak teslim etmek için yüksek performanslı bir yöntem sağlar. Bir ileti bir kuyruğa ya da konuya geldiğinde, JMS havuzdan bir iş parçacığı seçer ve ona bir ileti kümesi gönderir. Bunu yapmak için JMS, ilişkili bir MessageListener' ın `onMessage()` yöntemini çalıştırır.

Her biri kayıtlı bir MessageListenerile birden çok Oturum ve MessageConsumer nesnesi oluşturarak aynı etkiyi elde edebilirsiniz. Ancak, ConnectionConsumer , daha iyi performans, daha az kaynak kullanımı ve daha fazla esneklik sağlar. Özellikle, daha az oturum nesnesi gereklidir.

ASF ile bir uygulamanın planlanması

Bu bölümde aşağıdakiler de içinde olmak üzere bir uygulamanın nasıl planlanmanız gerektiğini anlatılıyor:

- “ASF kullanarak noktadan noktaya ileti sistemine ilişkin genel ilkeler” sayfa 890
- “ASF kullanarak ileti alışverişi yayınlama/abone olma genel ilkeleri” sayfa 891
- “ASF ' de kuyruktan iletilerin kaldırılması” sayfa 891
- ASF ' deki zehirli mesajların işlenmesi. Bkz. “Handling poison messages in IBM WebSphere MQ classes for JMS” sayfa 853.

ASF kullanarak noktadan noktaya ileti sistemine ilişkin genel ilkeler

ASF kullanarak noktadan noktaya ileti alışverişi hakkında genel bilgi için bu konuyu kullanın.

Bir uygulama QueueConnection nesnesinden bir ConnectionConsumer yarattığında, bir JMS kuyruğu nesnesi ve bir seçici dizgisi belirtir. ConnectionConsumer (ConnectionConsumer), ilişkili ServerSessionHavuzundaki oturumlara ileti sağlamaya başlar. İletiler kuyruğa gönderilir ve seçiciyle eşleşirlerse, ilişkili ServerSessionHavuzundaki oturumlara teslim edilir.

WebSphere MQ terimlerinde, kuyruk nesnesi yerel kuyruk yöneticisinde bir QLOCAL ya da QALIAS ile gönderme yapar. Bu bir QALIAS ise, QALIAS 'ın bir QLOCAL' a gönderme yapmalıdır. Tam olarak çözülmüş olan WebSphere MQ QLOCAL, *temeldeki QLOCAL*olarak bilinir. A ConnectionConsumer is said to be *etkin* if it is not closed and its parent QueueConnection is started.

Her biri farklı seçicilere sahip birden çok ConnectionConsumersiçin, temeldeki QLOCAL ile aynı şekilde çalıştırılacak şekilde kullanılabilir. Performansı korumak için, istenmeyen iletilerin kuyruğun üzerinde birikmemesi gerekir. İstenmeyen iletiler, etkin olmayan ConnectionConsumer ' ın eşleşen bir seçiciye sahip olduğu iletilerdir. QueueConnectionFactory 'yi, istenmeyen iletilerin kuyruktan kaldırılabilmesi için ayarlayabilirsiniz (ayrıntılar için bkz. “ASF ' de kuyruktan iletilerin kaldırılması” sayfa 891). Bu davranışı aşağıdaki iki yoldan birini kullanarak ayarlayabilirsiniz:

- Use the JMS administration tool to set the QueueConnectionFactory to MRET(NO).
- Programınızda şunu kullanın:

```
MQQueueConnectionFactory.setMessageRetention(WMQConstants.WMQ_MRET_NO)
```

Bu ayarı değiştirmezseniz, varsayılan değer bu tür istenmeyen iletilerin kuyruğun üzerinde tutulmasını sağlar.

WebSphere MQ kuyruk yöneticisini ayarladığınızda, aşağıdaki noktaları göz önünde bulundurun:

- Paylaşılan giriş için temeldeki QLOCAL etkinleştirilmelidir. Bunu yapmak için, aşağıdaki MQSC komutunu kullanın:

```
ALTER QLOCAL(your.qlocal.name) SHARE GET(ENABLED)
```

- Kuyruk yöneticinizin etkinleştirilmiş bir ölü harf kuyruğu olmalıdır. Bir ConnectionConsumer , bir iletiyi ölü-mektup kuyruğuna yerleştirirken bir sorunla karşılaştıysa, temel QLOCAL duraklarından ileti teslimi durur. Bir kuyruk-harf kuyruğu tanımlamak için aşağıdaki adresi kullanın:

```
ALTER QMGR DEADQ(your.dead.letter.queue.name)
```

- ConnectionConsumer çalıştıran kullanıcı, MQOO_SAVE_ALL_CONTEXT ve MQOO_PASS_ALL_CONTEXT ile MQOP gerçekleştirme yetkisine sahip olmalıdır. Ayrıntılar için, belirli bir altyapınıza ilişkin WebSphere MQ belgelerine bakın.
- Kuyruğun üzerinde istenmeyen iletiler bırakılırsa, bunlar sistem başarımını düşürmektedir. Therefore, plan your message selectors so that between them, the ConnectionConsumers will remove all messages from the queue.

MQSC komutlarıyla ilgili ayrıntılar için MQSC başvurusubaşlıklı konuya bakın.

ASF kullanarak ileti alışverişi yayınlama/abone olma genel ilkeleri

ConnectionConsumers , belirtilen bir Konu için ileti alır. ConnectionConsumer , dayanıklı ya da kalıcı olmayan bir değer olabilir. ConnectionConsumer tarafından kullanılan kuyruğu ya da kuyrukları belirlemeniz gerekir.

Bir uygulama TopicConnection nesnesinden bir ConnectionConsumer yarattığında, bir Konu nesnesini ve bir seçici dizgisini belirtir. ConnectionConsumer daha sonra, abone olunan konuya ilişkin alıkonan yayınlar da içinde olmak üzere, o Konu üzerindeki seçiciyle eşleşen iletileri almaya başlar.

Diğer bir seçenek olarak, bir uygulama belirli bir adla ilişkilendirilmiş kalıcı bir ConnectionConsumer (Bağlantı Tüketicisi) yaratabilir. This ConnectionConsumer receives messages that have been published on the Topic since the durable ConnectionConsumer was last active. Bu, Konu üzerindeki seçiciyle eşleşen tüm iletileri alır. Ancak, ConnectionConsumer okuma öbekini kullanıyorsa, istemci arabelleğindeki kalıcı olmayan iletileri kaybederse, bu ileti kapandığında kaybedilir.

JMS için WebSphere MQ sınıfları WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipiyse, kalıcı olmayan ConnectionConsumer abonelikleri için ayrı bir kuyruk kullanılır.TopicConnectionÜreticisi üzerindeki CCSUB yapılandırılabilir seçeneği, kullanılacak kuyruğu belirtir. Olağan durumda, CCSID ' ler aynı TopicConnectionüreticisini kullanan tüm ConnectionConsumers tarafından kullanılmak üzere tek bir kuyruk belirtir. Ancak, her bir ConnectionConsumer , ardından bir yıldız (*) işareti ve ardından bir kuyruk adı öneki belirtilerek geçici bir kuyruk oluşturmak mümkündür.

JMS için WebSphere MQ sınıfları WebSphere MQ ileti alışverişi sağlayıcısı geçiş kipinde olursa, Konudaki CCDSUB özelliği, kalıcı abonelikler için kullanılacak kuyruğu belirtir. Yine, varolan bir kuyruk ya da bir kuyruk adı öneki ve ardından bir yıldız işareti (*) kullanılabilir. Önceden var olan bir kuyruğu belirlerseniz, Konu 'a abone olan tüm dayanıklı ConnectionConsumers bu kuyruğu kullanır. Bir kuyruk adı öneki ve ardından bir yıldız işareti (*) belirtirseniz, bir kuyruk ilk kez, belirli bir adla yaratılan kalıcı bir ConnectionConsumer ' nin yaratıldığını belirtir. Bu kuyruk, daha sonra aynı adda bir kalıcı ConnectionConsumer yaratılırsa yeniden kullanılır.

WebSphere MQ kuyruk yöneticisini ayarladığınızda, aşağıdaki noktaları göz önünde bulundurun:

- Kuyruk yöneticinizin etkinleştirilmiş bir ölü harf kuyruğu olmalıdır. Bir ConnectionConsumer , bir iletiyi ölü-mektup kuyruğuna yerleştirirken bir sorunla karşılaştıysa, temel QLOCAL duraklarından ileti teslimi durur. Bir kuyruk-harf kuyruğu tanımlamak için aşağıdaki adresi kullanın:

  ```
  ALTER QMGR DEADQ(your.dead.letter.queue.name)
  ```

- ConnectionConsumer çalıştıran kullanıcı, MQOO_SAVE_ALL_CONTEXT ve MQOO_PASS_ALL_CONTEXT ile MQOP gerçekleştirme yetkisine sahip olmalıdır. Ayrıntılar için, altyapınıza ilişkin WebSphere MQ belgelerine bakın.
- Ayrı, adanmış bir kuyruk oluşturarak, tek bir ConnectionConsumer için performansı en iyi duruma getirebilirsiniz. Bu, fazladan kaynak kullanımı maliyetidir.

ASF ' de kuyruktan iletilerin kaldırılması

Bir uygulama ConnectionConsumers'ı kullandığında, JMS' nin bazı durumlarda kuyruktan ileti çıkarması gerekebilir.

Bu durumlar aşağıdaki gibidir:

Hatalı biçimlendirilmiş ileti
JMS ' nin ayrıştıramayacağı bir ileti gelebilir.

Zehir mesajı
Bir ileti geriletme eşiğine ulaşabilir, ancak ConnectionConsumer bunu, geriletme kuyruğunda istekte bulunmaya başarısız olur.

İlgili ConnectionConsumeryok
Noktadan noktaya ileti alışverişi için, QueueConnectionFactory istenmeyen iletileri alıkoymayacak şekilde ayarlandığında, ConnectionConsumerstarafından istenmeyen bir ileti gönderilir.

Bu durumlarda, ConnectionConsumer iletiyi kuyruktan kaldırma girişiminde bulunur. İletinin MQMD ' nin rapor alanındaki yok etme seçenekleri tam davranışı ayarlar. Bu seçenekler şunlardır:

MQRO_DEAD_LETTER_Q
İleti, kuyruk yöneticisinin ölü harf kuyruğu için istekte bulunmuyor. Bu varsayılandır.

MQRO_DISCARD_MSG
İleti atılır.

ConnectionConsumer bir rapor iletisi de oluşturur ve bu ileti, iletinin MQMD ' nin rapor alanına da bağlıdır. This message is sent to the message's ReplyToQ on the ReplyToQmgr. Rapor iletisi gönderilirken bir hata varsa, ileti, bunun yerine ölü-mektup kuyruğuna gönderilir. İletinin MQMD kümesi ayrıntılarının rapor alanındaki kural dışı durum raporu seçenekleri, rapor iletisinin ayrıntılarına ayarlanır. Bu seçenekler şunlardır:

MQRO_EXCEPTION
Özgün iletinin MQMD ' yi içeren bir rapor iletisi oluşturulur. Herhangi bir ileti gövdesi verisi içermiyor.

MQRO_EXCEPTION_WITH_DATA
MQMD ' yi, herhangi bir MQ üstbilgilerini ve 100 baytlık gövde verilerini içeren bir rapor iletisi oluşturulur.

MQRO_EXCEPTION_WITH_FULL_DATA
Özgün iletiden tüm verileri içeren bir rapor iletisi oluşturulur.

varsayılan
Rapor iletisi oluşturulmadı.

Rapor iletileri oluşturulduğunda, aşağıdaki seçenekler onurlandırılır:

- MQRO_NEW_MSG_ID
- MQRO_PASS_MSG_ID
- MQRO_COPY_MSG_ID_TO_CORREL_ID
- MQRO_PASS_COREL_ID

Bir zehir iletisi yeniden istenemezse, belki de ölü-mektup kuyruğu tam ya da yetki yanlış belirtilmiş olduğundan, iletinin kalıcılısına bağlıdır. İleti kalıcı değilse, ileti atılır ve herhangi bir rapor iletisi oluşturulmadı. İleti kalıcıysa, iletilerin o hedef durakları dinleyerek tüm bağlantı tüketicilerine teslim edilir. Bu tür bağlantı tüketicileri kapatılmalı ve sorun, yeniden oluşturulabilmesi ve ileti tesliminin yeniden başlatılabilmesi için çözülmeden önce çözülmelidir.

Herhangi bir sorun oluşmadığından emin olmak için, bir ölü-mektup kuyruğu tanımlanması ve düzenli olarak kontrol etmek önemlidir. Özellikle, ölü harf kuyruğunun derinlik üst sınırına ulaşmadığından ve ileti boyutu üst sınırının tüm iletiler için yeterince büyük olduğundan emin olun.

When a message is requeued to the dead-letter queue, it is preceded by a WebSphere MQ dead-letter header (MQDLH). MQDLH biçimiyle ilgili ayrıntılar için MQDLH-Dead-letter header başlıklı konuya bakın. Bir ConnectionConsumer 'in ölü-mektup kuyruğuna yerleştirdiği ya da bir ConnectionConsumer ' in oluşturduğu rapor iletilerini aşağıdaki alanlar tarafından tanımlayabilir:

- PutApplTipi `MQAT_JAVA (0x1C)`
- PutApplAdı "`MQ JMS ConnectionConsumer`"

Bu alanlar, ölülerin kuyruğunda bulunan iletilerin MQDLH 'de ve rapor iletilerinin MQMD' de yer alıyor. MQMD 'nin geri bildirim alanı ve MQDLH' nin neden alanı, hatayı açıklayan bir kod içerir. Bu kodlarla ilgili ayrıntılı bilgi için bkz. “ASF ' deki neden ve geri bildirim kodları” sayfa 893. Diğer alanlar, MQDLH-Dead-letter üstbilgisiiçinde anlatıldığı gibidir.

ASF ' de zehirli iletilerin işlenmesi

Uygulama Sunucusu Tesisi içinde, zehirli ileti işleme, JMS için WebSphere MQ sınıflarındaki başka bir yerde biraz farklı şekilde işlenir.

JMS için WebSphere MQ sınıflarında zehirli ileti işleme hakkında bilgi için bkz. “Handling poison messages in IBM WebSphere MQ classes for JMS” sayfa 853.

Application Server Facilitis (ASF) olanağını kullandığınızda, MessageConsumeryerine ConnectionConsumer, zehirli iletileri işler. ConnectionConsumer , kuyruğun BackoutThreshold ve BackoutRequeueQName özelliklerine göre iletilmekte olan iletileri sağlar.

Bir uygulama ConnectionConsumers' ı kullandığında, bir iletinin yedekleneceği koşullar, uygulama sunucusunun sağladığı oturuma bağlıdır:

- Oturum, AUTO_RELSE ya da DUP_OK_RELSE ile birlikte işlem dışı olduğunda, bir ileti yalnızca sistem hatasından sonra ya da uygulama beklenmedik bir şekilde sona erdirildikten sonra yedeklenir.
- When the session is non-transacted with CLIENT_ACKNOWLEDGE, unacknowledged messages can be backed out by the application server calling Session.recover().

  Typically, the client implementation of MessageListener or the application server calls Message.acknowledge(). Message.acknowledge() acknowledges all messages delivered on the session so far.
- When the session is transacted, unacknowledged messages can be backed out by the application server calling Session.rollback().
- Uygulama sunucusu bir XASSession sağladıysa, iletiler dağıtılmış bir harekete bağlı olarak kesinleştirilir ya da yedeklenir. Uygulama sunucusu, işlemin tamamlanması için sorumluluk alır.

WebSphere Application Server, Sürüm 5.0 ve Sürüm 5.1 içindeki yerleşik JMS sağlayıcısı, JMS için WebSphere MQ sınıfları için açıklanan, farklı bir şekilde zehirli iletileri işler. Yerleşik JMS sağlayıcısının zehirli iletileri nasıl işleyeceğini hakkında bilgi için, ilgili WebSphere Application Server ürün belgelerine bakın.

Hata işleme

Bu bölümde, “ASF ' deki hata koşullarından kurtarma” sayfa 893 ve “ASF ' deki neden ve geri bildirim kodları” sayfa 893de içinde olmak üzere, hata işlenmesinin çeşitli yönleri yer alıyor.

ASF ' deki hata koşullarından kurtarma

Bir ConnectionConsumer , ciddi bir hata ortaya çıkardığında, aynı QLOCAL duraklarına ilgi içeren tüm ConnectionConsumers iletisine ileti teslim eder. Bu gerçekleştiğinde, etkilenen bağlantıya kayıtlı olan tüm ExceptionListener ' ler bildirim gönderilir. Bir uygulamanın bu hata koşullarından kurtulması için iki yol vardır.

Tipik olarak, ConnectionConsumer , bir iletiyi ölü-mektup kuyruğuna gönderemezse ya da QLOCAL ' tan ileti okurken hata ortaya çıktığında bu tür ciddi bir hata oluşur.

Etkilenen bağlantıya kayıtlı herhangi bir ExceptionListener (ExceptionListener) bildirildiğinden, sorunun nedenini belirlemek için bunları kullanabilirsiniz. Bazı durumlarda, sistem yöneticisinin sorunu çözmek için müdahale etmesi gerekir.

Bu hata koşullarından kurtulmak için aşağıdaki tekniklerden birini kullanın:

- Etkilenen tüm ConnectionConsumers'ta `close()` ' u arayın. Uygulama yeni ConnectionConsumers ' ı ancak tüm etkilenen ConnectionConsumers kapatıldıktan sonra yaratabilir ve sistem sorunları çözülmüş olur.
- Etkilenen tüm Connections 'da `stop()` ' u arayın. Tüm Connections durdurulduktan ve herhangi bir sistem sorunu çözüldükten sonra, uygulama Connections 'ı başarıyla `start()` ' e verebilir.

ASF ' deki neden ve geri bildirim kodları

Bir hatanın nedenini belirlemek için neden ve geri bildirim kodlarını kullanın. ConnectionConsumer tarafından oluşturulan ortak neden kodları burada verilmiştir.

Bir hatanın nedenini belirlemek için aşağıdaki bilgileri kullanın:

- Herhangi bir rapor iletisinde geribildirim kodu
- Ölü-mektup kuyruğunda herhangi bir iletinin MQDLH ' de neden kodu

ConnectionConsumers , aşağıdaki neden kodlarını oluşturur.

MQRC_BACKOUT_THRESHOLD_ULAS (0x93A; 2362)

Neden
İleti, QLOCAL üzerinde tanımlanan Arka Uç Eşiğine ulaştı, ancak Geri Dönme Kuyruğu tanımlanmadı.

Backout Kuyruğu tanımlayamadığınız platformlarda, ileti 20 'nin JMS tanımlı geriletme eşiğine ulaştı.

İşlem
Bu istenmiyorsa, ilgili QLOCAL için Geri Kayan Kuyruğu tanımlayın. Ayrıca, birden çok geri tepenin nedenini de arayın.

MQRC_MSG_NOT_MATCHED (0x93B; 2363)

Neden
Noktadan noktaya ileti sisteminde, kuyruğu izleyen ConnectionConsumers için seçicilerden hiçbiriyle eşleşmeyen bir ileti vardır. Performansı korumak için, ileti, ölü-mektup kuyruğuna istekte bulunmaya devam eder.

İşlem
To avoid this situation, ensure that ConnectionConsumers using the queue provide a set of selectors that deal with all messages, or set the QueueConnectionFactory to retain messages.

Diğer bir seçenek olarak, iletinin kaynağını araştırın.

MQRC_JMS_FORMAT_ERROR (0x93C; 2364)

Neden
JMS, kuyruktaki iletiyi yorumlayamıyor.

İşlem
İletinin kökenini araştırın. JMS genellikle BytesMessage ya da TextMessageolarak beklenmeyen bir biçime ilişkin iletiler sağlar. Bazen, ileti çok kötü biçimlendirilirse bu hata başarısız olur.

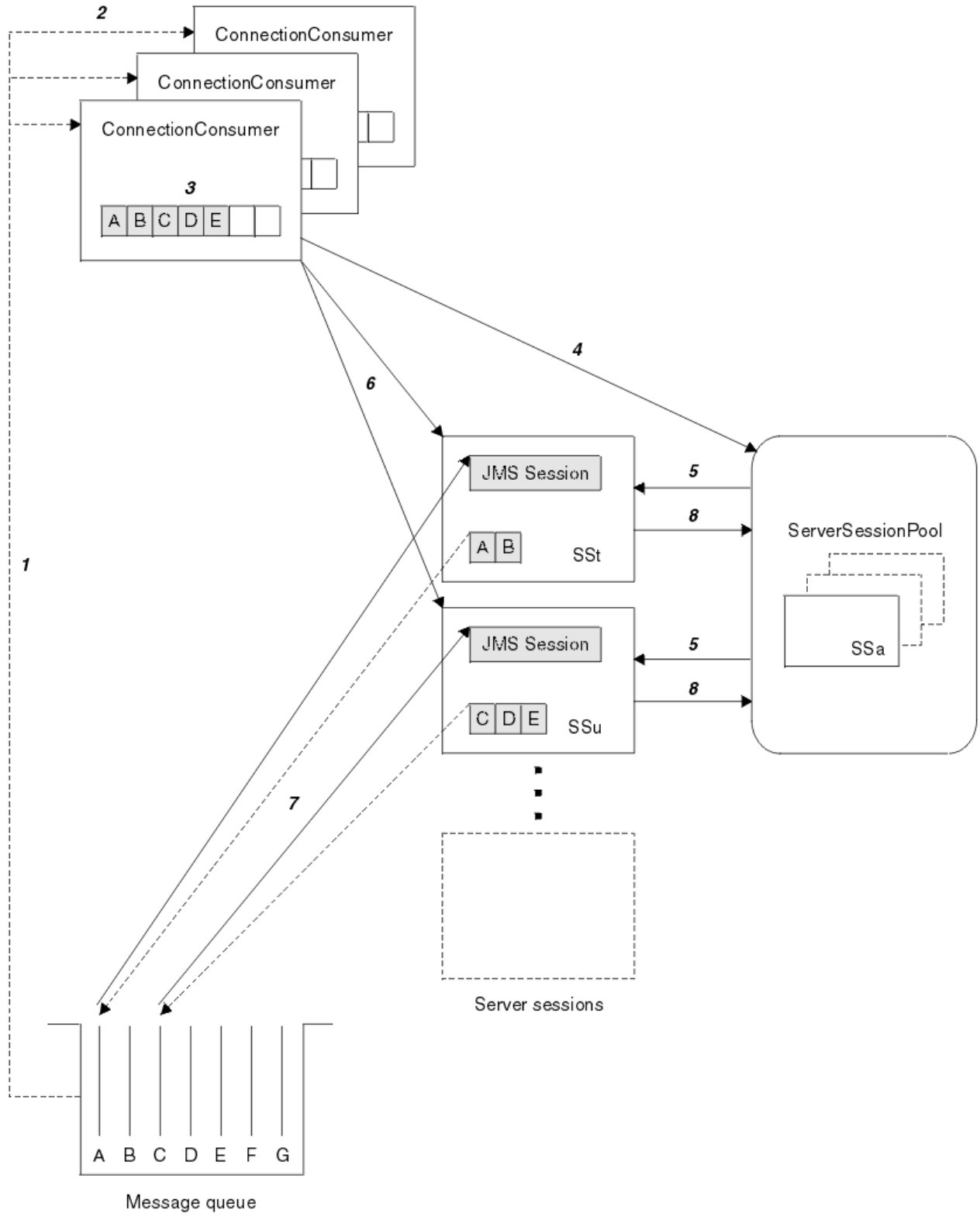
Bu alanlarda görüntülenen diğer kodların nedeni, başarısız olan bir iletiyi geri alma kuyruğuna yeniden istekte bulunmaya çalışmalarından kaynaklanır. Bu durumda, kodda, istekte bulunanın başarısız olmasının nedeni açıklanmaktadır. Bu hataların nedenini tanılamak için API neden kodlarıkonusuna bakın.

Rapor iletisi ReplyToQ ' ya konulamazsa, bu ileti, ölü-mektup kuyruğuna konabilecektir. Bu durumda, MQMD ' nin geri bildirim alanı bu konuda açıklandığı gibi tamamlanır. MQDLH 'daki neden alanı, rapor iletisinin ReplyToQ' ya neden yerleştirilememesini açıklar.

The function of a server session pool in AFS

Bu konuda, bir sunucu oturumu havuzunun işlevi özetlenir.

Şekil 165 sayfa 895 , ServerSessionhavuzu ve ServerSession işlevselliğinin ilkelerini özetler.



Şekil 165. ServerSessionhavuzu ve ServerSession işlevi

1. ConnectionConsumers , kuyruktan ileti başvurularını alır.
2. Her ConnectionConsumer , belirli ileti başvurularını seçer.
3. ConnectionConsumer arabelleği, seçilen ileti başvurularını bulundurur.
4. ConnectionConsumer , ServerSessionHavuzundan bir ya da daha fazla ServerSessions isteğinde bulunur.

5. ServerSessions , ServerSessionHavuzundan ayrılır.
6. ConnectionConsumer , ServerSessions ' a ileti başvuruları atar ve çalışmakta olan ServerSession iş parçacığını başlatır.
7. Her ServerSession , gönderme yapılan iletileri kuyruktan alır. Bu, bunları JMS oturumuyla ilişkili MessageListener ' dan onMessage yöntemine geçirir.
8. İşlemin tamamlanmasından sonra, ServerSession havuza geri döndürülür.

Bir uygulama sunucusu, olağan durumda ServerSessionhavuzu ve ServerSession işlevlerini sağlar.

WebSphere MQ JMS yönetim aracının kullanılması

JMS nesnesi için sekiz WebSphere MQ sınıflarının özelliklerini tanımlamak ve bunları bir JNDI ad alanı içinde saklamak için yönetim aracını kullanın. Daha sonra uygulamalar, bu yönetilen nesneleri ad alanından almak için JNDI olanağını kullanabilir.

The WebSphere MQ classes JMS objects that you can administer by using the tool are:

- MQConnectionFactory
- MQQueueConnectionÜreticisi
- MQTopicConnectionÜreticisi
- MQQueue
- MQTopic
- MQXAConnectionFactory
- MQXAQueueConnectionÜreticisi
- MQXATopicConnectionÜreticisi

Bu nesnelerle ilgili ayrıntılar için bkz. “JMS nesnelerinin yönetilmesi” sayfa 900 .

Bu aracı kullanmak için gerek duyduğunuz özellik tipleri ve değerleri IBM WebSphere MQ classes for JMS nesnelerinin özelliklerilistesinde yer almektedir.

Araç ayrıca, denetimcilerin JNDI içindeki dizin ad alanı alt bağlamlarını değiştirmelerine de olanak sağlar. Bkz. “Alt bağlamların WebSphere MQ JMS yönetim aracı ile kullanılması” sayfa 900.

You can also create and configure JMS administered objects with the WebSphere MQ Explorer.

IBM WebSphere MQ classes for JMS denetim aracını çağırma

Yönetim aracının komut satırı arabirimi vardır. Bunu etkileşimli olarak kullanabilir ya da toplu iş işlemi başlatmak için kullanabilirsiniz.

Etkileşimli kip, denetim komutlarını girebileceğiniz bir komut istemi sağlar. Toplu kipte, aracı başlatmak için kullanılan komut, denetim komut dosyası içeren bir dosyanın adını içerir.

Etkileşimli kip

Aracı etkileşimli kipte başlatmak için şu komutu girin:

```
JMSAdmin [-t] [-v] [-cfg config_filename]
```

Burada:

-t
İzlemeyi etkinleştirir (varsayılan izleme kapalıdır)

The trace file is generated in "%MQ_JAVA_DATA_PATH%"\errors (Windows) or /var/mqm/trace (UNIX). İzleme dosyasının adı şu biçimden olur:

mqjms_*PID*.trc

Burada *PID* , JVM ' nin işlem tanıtıcısıdır.

-v
Ayrıntılı çıkış üretir (varsayılan terse çıkışıdır)

-cfg config_dosyaadı
Alternatif bir yapılandırma dosyasını adlandırır. Bu parametre atılırsa, varsayılan yapılandırma dosyası olan JMSAdmin.configkullanılır. (Bkz. “JMS Denetimi aracının yapılandırılması” sayfa 897)

Aracın yönetim komutlarını kabul etmeye hazır olduğunu belirten bir komut istemi görüntülenir. Bu bilgi istemi başlangıçta şu şekilde görünür:

```
InitCtx>
```

yürürlükteki bağlamın (yani, tüm adlandırma ve dizin işlemlerinin şu anda gönderme yaptığı JNDI bağlamının), PROVIDER_URL yapılanış değiştirgesinde (bkz. “JMS Denetimi aracının yapılandırılması” sayfa 897) tanımlanan başlangıç bağlamının olduğunu belirtir.

Dizin ad alanını geçerken, bilgi istemi bunu yansıtacak şekilde değişir, böylece bilgi istemi her zaman geçerli bağlamı görüntüler.

Toplu iş kipi

Aracı toplu kipte başlatmak için şu komutu girin:

```
JMSAdmin <test.scp
```

Burada *test.scp* , denetim komutlarını içeren bir komut kütüğüdür (bkz. “WebSphere MQ JMS yönetim aracındaki yönetim komutları” sayfa 899). Kütükteki son komut END komutu olmalıdır.

JMS Denetimi aracının yapılandırılması

WebSphere MQ JMS Denetimi aracı, belirli özelliklerin değerlerini ayarlamak için bir yapılandırma dosyası kullanır. Sisteminize uyarlayabileceğiniz örnek bir dosya sağlanır.

Yapılandırma dosyası, eşittir işaretiyle (=) ayrılmış, bir anahtar değer çiftinden oluşan düz metin dosyasıdır. Bu, aşağıdaki örnekteki gösterilmektedir:

```
#Set the service provider
    INITIAL_CONTEXT_FACTORY=com.sun.jndi.ldap.LdapCtxFactory
#Set the initial context
    PROVIDER_URL=ldap://polaris/o=ibm_us,c=us
#Set the authentication type
    SECURITY_AUTHENTICATION=none
```

(çizginin ilk sütunundaki bir #, bir yorumu ya da kullanılmayan bir çizgiyi belirtir.)

Örnek bir yapılandırma dosyası WebSphere MQile birlikte sağlanır. Dosya, `JMSAdmin.config`dizininde bulunur ve bu dosya `<MQ_JAVA_INSTALL_PATH>/bin` dizininde bulunur. Bu dosyayı, sisteminizin ayarına uygun olacak şekilde düzenleyin.

Yönetim aracını aşağıdaki özellikler için değerlerle yapılandırın:

INITIAL_CONTEXT_FACTORY
Aracın kullandığı hizmet sağlayıcı. Bu özellik için desteklenen değerler aşağıdaki gibidir:

- com.sun.jndi.ldap.LdapCtxFactory (LDAP için)
- com.sun.jndi.fscontext.RefFSContextFactory (dosya sistemi bağlamı için)

Önceki listede olmayan bir InitialContext(InitialContext) üreticisini de kullanabilirsiniz. Daha fazla ayrıntı için bkz. “Listelenmeyen bir InitialContextÜreticisi 'ni WebSphere MQ JMS yönetim aracı ile kullanma” sayfa 898 .

PROVIDER_URL
Oturumun ilk bağlamının URL 'si; araç tarafından gerçekleştirilen tüm JNDI işlemlerinin kökü. Bu özelliğin iki biçimi desteklenir:

- ldap://hostname/contextname
- dosya: [sürücü:] /yol adı

LDAP URL adresinin biçimi, LDAP sağlayıcısına bağlı olarak değişiklik gösterebilir. Ek bilgi için LDAP belgelerinize bakın.

GÜVENLIK_KIMLIK
JNDI, güvenlik kimlik bilgilerini hizmet sağlayıcısına iletir. Bu özellik, yalnızca bir LDAP hizmet sağlayıcısı kullanıldığında kullanılır. Bu " zellik, üç de§erden birini alabilir:

- none (anonim kimlik doğrulama)
- basit (basit kimlik doğrulama)
- CRAM-MD5 (CRAM-MD5 kimlik doğrulama mekanizması)

Geçerli bir değer sağlanmazsa, özellik varsayılan olarak none değerini alır. Yönetim aracından güvenliğe ilişkin ayrıntılar için bkz. “JMS yönetim aracı için güvenliğin yapılandırılması” sayfa 898 .

Bu özellikler, bir yapılandırma dosyasında ayarlanır. Aracı çağırdığınızda, “IBM WebSphere MQ classes for JMS denetim aracını çağırma” sayfa 896içinde açıklandığı gibi `-cfg` komut satırı parametresini kullanarak bu yapılandırmayı belirleyebilirsiniz. Bir yapılandırma dosyası adı belirtmezseniz, araç varsayılan yapılandırma dosyasını yüklemeye çalışır (`JMSAdmin.config`). Bu dosya, önce yürürlükteki dizinde, sonra da `<MQ_JAVA_INSTALL_PATH>/bin` dizininde bu dosyayı arar; burada `<MQ_JAVA_INSTALL_PATH>` , JMS kuruluşu için WebSphere MQ sınıflarınızın yoludur.

Listelenmeyen bir InitialContextÜreticisi 'ni WebSphere MQ JMS yönetim aracı ile kullanma

İki InitialContextüretici değerleri desteklenir. JMS denetimi yapılandırma dosyasında parametreleri ayarlayarak diğer JNDI bağlamlarını kullanabilirsiniz.

Yönetim aracını kullanarak, JMSAdmin yapılandırma dosyasında tanımlı olan üç değiştirge kullanarak, “JMS Denetimi aracının yapılandırılması” sayfa 897 içinde listelenenler dışındaki JNDI bağlamlarına bağlanabilirsiniz.

Farklı bir InitialContextüreticiyi kullanmak için:

1. INITIAL_CONTEXT_FACTORY özelliğini, gerekli sınıf adına ayarlayın.
2. USE_INITIAL_DIR_CONTEXT, NAME_PREFIX ve NAME_READABILY_MARKER özelliklerini kullanarak InitialContextFactory 'nin davranışını tanımlayın.

Bu özelliklere ilişkin ayarlar, örnek yapılandırma dosyası açıklamalarında açıklanmaktadır.

Desteklenen INITIAL_CONTEXT_FACTORY değerlerinden birini kullanırsanız, burada listelenen üç özelliği tanımlamanıza gerek yoktur. Ancak, sistem varsayılanlarını geçersiz kılmak için bu değerleri verebilirsiniz. Üç InitialContextÜreticisi özelliklerinden birini ya da birkaçını çıkarırsanız, denetim aracı diğer özelliklerin değerlerine dayalı olarak uygun varsayılan değerler sağlar.

JMS yönetim aracı için güvenliğin yapılandırılması

Güvenlik kimlik bilgilerinin hizmet sağlayıcıya iletilip iletilmediğini belirlemek için SECURITY_AUTHENTICATION özelliğini kullanın.

SECURITY_AUTHENTICATION özelliği, “JMS Denetimi aracının yapılandırılması” sayfa 897içinde açıklanmıştır. Bunun etkisi aşağıdaki gibidir:

- Bu parametreyi `none`olarak ayarladıysanız, JNDI hizmet sağlayıcıya hiçbir güvenlik kimlik bilgisi iletmez ve *anonim kimlik doğrulaması* gerçekleştirilir.
- Parametreyi `basit` ya da `CRAM-MD5`olarak ayarlıyorsanız, güvenlik kimlik bilgileri temeldeki hizmet sağlayıcıya JNDI aracılığıyla geçirilir. Bu güvenlik kimlik bilgileri, kullanıcı ayırt edici adı (Kullanıcı DN 'si) ve parola biçiminde bulunur.

Güvenlik kimlik bilgileri gerekliyse, araç başlatıldığında sizden bilgi isteminde bulunduğunuzda bilgi isteminde bulunduğunuzda JMSAdmin yapılandırma dosyasında PROVIDER_USERDN ve PROVIDER_PASSWORD özelliklerini ayarlayarak bunu önlein.

Not: Bu özellikleri kullanmayacaksa, metin yazılan *parola dahil*metni ekrana yansıtılır. Bunun güvenlik sonuçları olabilir.

Araç kimlik doğrulaması yapmaz; görev, LDAP sunucusuna devredilir. LDAP sunucusu yöneticisi, dizinin farklı bölümlerine erişim ayrıcalıklarını kurmalı ve sürdürmelidir. Ek bilgi için LDAP belgelerinize bakın. Kimlik doğrulaması başarısız olursa, araç uygun bir hata iletisi görüntüler ve sonlandırılır.

Güvenlik ve JNDI ile ilgili daha ayrıntılı bilgi, Sun 'ın Java web sitesinde (https://java.sun.com) yer alan belgelerdir.

WebSphere MQ JMS yönetim aracındaki yönetim komutları

Yönetim aracı, bir denetim komutu ve uygun değiştirgelerinden oluşan komutları kabul eder.

Komut istemi görüntülendiğinde, araç komutları kabul etmeye hazır olur. Denetim komutları genellikle aşağıdaki biçimden olur:

```
verb [param]*
```

Burada **verb** , Çizelge 133 sayfa 899içinde listelenen yönetim fiillerinden biridir. Geçerli tüm komutlar, komutun başlangıcındaki standart ya da kısa biçimdeki bir fiil içerir.

Bir fiilin değiştirebileceği parametreler, yüke bağlıdır. Örneğin, END komutu herhangi bir parametre alamıyor, ancak DEFINE yüklemi herhangi bir sayıda parametreyi alabilir. İlgili konularda en az bir parametre alan fiillerin ayrıntıları ele alınmıştır.

Çizelge 133. Yönetim fiilleri

Komut	**Kısa Biçim**	**Tanım**
ALTER	Alt	Denetimli nesne özelliklerinin en az birini değiştirir
Tanımla	DÖF	Yönetilen bir nesne yaratın ve saklayın ya da bir alt bağlam yaratın
GÖRÜNTÜLE	DIS	Saklanmış denetlenen bir ya da daha çok nesnenin ya da yürürlükteki bağlamın içindekileri görüntüler
SİL	Sil	Ad alanından bir ya da daha çok yönetilen nesneyi kaldırın ya da boş bir alt bağlamı kaldırın
Değiştir	CHG	Yürürlükteki bağlamı değiştirerek, kullanıcının dizin ad alanını ilk bağlamın altında herhangi bir yerde geçmesine izin verir (güvenlik açıklığı beklemede).
Kopyala	CP	Saklanmış bir yönetilen nesnenin bir kopyasını yapın ve bunu alternatif bir ad altında saklayan
Taşı	MV	Denetlenen bir nesnenin altında saklandığı adı değiştirin
BİTİŞ		Denetim aracını kapat

Komut adları büyük ve küçük harfe duyarlı değildir.

Genellikle, komutları sonlandırmak için satır başı tuşuna basmanız gerekir. Ancak, satır başı döndürmeden önce artı işaretini (+) yazarak bu işlemi geçersiz kılabilirsiniz. Bu, aşağıdaki örnekte gösterildiği gibi çok satırlı komutlar girmenizi sağlar:

```
DEFINE Q(BookingsInputQueue) +
       QMGR(QM.POLARIS.TEST) +
       QUEUE(BOOKINGS.INPUT.QUEUE) +
```

```
        PORT(1415) +
        CCSID(437)
```

Aşağıdaki karakterlerden herhangi biriyle başlayan satırlar açıklama olarak ele alınır ve yoksayılır: * #/.

Alt bağlamların WebSphere MQ JMS yönetim aracı ile kullanılması

Dizin ad alanı alt bağlamlarını işlemek için **CHANGE**, **DEFINE**, **DISPLAY** ve **DELETE** fiillerini kullanın.

Bu fiillerin kullanımı, Çizelge 134 sayfa 900içinde açıklanmaktadır.

Çizelge 134. Alt bağlamları işlemek için kullanılan komutların sözdizimi ve tanımı

Komut sözdizimi	**Tanım**
DEFINE CTX (ctxName)	Geçerli bağlamın alt bağlamını yaratma girişimleri (ctxNameadını taşıyan). Bir güvenlik ihlali varsa, alt bağlamın önceden varsa ya da sağlanan ad geçerli değilse başarısız olur.
CTX	Geçerli bağlamın içeriğini görüntüler. Denetlenen nesnelere aile ek açıklama konması, `[D]`ile alt bağlamlar olarak eklemesi. Her nesneye ilişkin Java tipi de görüntülenir.
DELETE CTX (ctxName)	Yürürlükteki bağlamın alt bağlamını (ctxName) silme girişiminde bulunun. Bağlam bulunamazsa, boş değilse ya da bir güvenlik ihlali varsa başarısız olur.
DEğIşTIR CTX (ctxName)	Yürürlükteki bağlamı, şimdi ctxNameadını taşıyan alt bağlamla gönderme yapıyor. ctxName özel değerlerinden biri sağlanabilir: **= AÇIK** Yürürlükteki bağlamın üst öğesine gider **= INIT** doğrudan başlangıç bağlamına taşır Belirtilen bağlam yoksa ya da bir güvenlik ihlali olduğunda başarısız olur.

JMS nesnelerinin yönetilmesi

Bu bölümde, denetim aracının işleyebileceği sekiz nesne tipi açıklanmaktadır. Bu ürün, yapılandırılabilir özelliklerinin her biriyle ilgili ayrıntıları ve bunları işleyebilen fiillerle ilgili ayrıntıları içerir.

You can also create and configure JMS administered objects with the WebSphere MQ Explorer.

JMS nesne tipleri

Çizelge, denetlenen nesnelerin sekiz tipini gösterir.

Anahtar Sözcük sütunu, Çizelge 136 sayfa 902' ta gösterilen komutlarda *TYPE* yerine koyabileceğiniz dizgileri gösterir.

Çizelge 135. Yönetim aracı tarafından işlenen JMS nesnesi tipleri

Nesne Tipi	**anahtar sözcük**	**Tanım**
MQConnectionFactory	CF	JMS ConnectionFactory arabirimindeki WebSphere MQ somutlaması. Bu, hem noktadan noktaya iletişim, hem de yayınlama/abone olma etki alanlarında bağlantı yaratmak için kullanılan bir üretici nesnesini gösterir.

Çizelge 135. Yönetim aracı tarafından işlenen JMS nesnesi tipleri (devamı var)

Nesne Tipi	**anahtar sözcük**	**Tanım**
MQQueueConnectionÜreticisi	QCF	JMS QueueConnectionFactory arabiriminin WebSphere MQ somutlaması. Bu, noktadan noktaya iletişim etki alanında bağlantı yaratılmasına ilişkin bir üretici nesnesini gösterir.
MQTopicConnectionÜreticisi	TCF	JMS TopicConnectionFactory arabiriminin WebSphere MQ somutlaması. Bu, yayınlama/abone olma etki alanında bağlantılar yaratmak için kullanılan bir üretici nesnesini gösterir.
MQQueue	Q	JMS Kuyruğu arabiriminin WebSphere MQ somutlaması. Bu, noktadan noktaya iletişim alanındaki iletiler için bir hedef gösterir.
MQTopic	T	JMS Konu arabiriminin WebSphere MQ somutlaması. Bu, yayınlama/abone olma etki alanındaki iletiler için bir hedef gösterir.
MQXAConnectionFactory[“1” sayfa 901]	XACF	JMS XAConnectionFactory arabiriminin WebSphere MQ somutlaması. Bu, hem noktadan noktaya, hem de yayınlama/abone olma etki alanlarında ve bağlantıların JMS sınıflarının XA sürümlerini kullanan bağlantıları yaratmak için kullanılan bir üretici nesnesini gösterir.
MQXAQueueConnectionÜreticisi[“1” sayfa 901]	XAQCF	JMS XAQueueConnectionFactory arabiriminin WebSphere MQ somutlaması. Bu, JMS sınıflarının XA sürümlerini kullanan noktadan noktaya iletişim etki alanında bağlantı yaratılmasına ilişkin bir üretici nesnesini gösterir.
MQXATopicConnectionÜreticisi[“1” sayfa 901]	XATCF	JMS XATopicConnectionFactory arabiriminin WebSphere MQ somutlaması. Bu, JMS sınıflarının XA sürümlerini kullanan yayınlama/abone olma etki alanında bağlantı yaratmak için kullanılan bir üretici nesnesini gösterir.

Not:

1. Bu sınıflar, uygulama sunucuları satıcıları tarafından kullanılmak üzere sağlanır. Uygulama programcıları için doğrudan yararlı olma olasıkları düşük.

JMS nesneleriyle kullanılan fiiller

Yönetilen nesneleri dizin ad alanı içinde işlemek için `ALTER`, `DEFINE`, `DISPLAY`, `DELETE`, `COPY`ve `MOVE` fiillerini kullanabilirsiniz.

Çizelge 136 sayfa 902 , bu fiillerin kullanımını özetler. Substitute *TYPE* with the keyword that represents the required administered object, as listed in Çizelge 135 sayfa 900.

Çizelge 136. Yönetilen nesneleri işlemek için kullanılan komutların sözdizimi ve tanımı	
Komut sözdizimi	**Tanım**
ALTER *TYPE*(ad) [property] *	Denetlenen nesnenin özelliklerini sağlanan nesnelerle güncelleme girişiminde bulunanlar. Bir güvenlik ihlali varsa, belirtilen nesne bulunamazsa ya da sağlanan yeni özellikler geçerli değilse başarısız olur.
DEFINE *TIP*(ad) [özellik] *	Sağlanan özelliklerle *TYPE* tipinde bir yönetilen nesne yaratma ve bu nesneyi geçerli bağlamda name adının altında saklamanız için girişimde bulunur. Bir güvenlik ihlali varsa, belirtilen ad geçerli değilse ya da bu adı içeren bir nesne varsa ya da sağlanan özellikler geçersiz ise başarısız olur.
DISPLAY *TIP*(ad)	Displays the properties of the administered object of type *TYPE*, bound under the name name in the current context. Nesne yoksa ya da bir güvenlik ihlali olduğunda başarısız olur.
DELETE *TIP*(ad)	Attempts to remove the administered object of type *TYPE*, having the name name, from the current context. Nesne yoksa ya da bir güvenlik ihlali olduğunda başarısız olur.
TIP(nameA) *TIP*(nameB)	Makes a copy of the administered object of type *TYPE*, having the name nameA, naming the copy nameB. Bunların tümü yürürlükteki bağlamın kapsamı içinde gerçekleşir. Kopyalanacak nesne yoksa, nameB adlı bir nesne varsa ya da bir güvenlik ihlali olduğunda hata ortaya çıktı.
MOVE *TIP*(nameA) *TIP*(nameB)	Moves (renames) the administered object of type *TYPE*, having the name nameA, to nameB. Bunların tümü yürürlükteki bağlamın kapsamı içinde gerçekleşir. Taşınmak için nesne yoksa, nameB adlı bir nesne varsa ya da bir güvenlik ihlali varsa, başarısız olur.

WebSphere MQ JMS yönetim aracı ile nesne yaratılması

Nesneleri yaratın ve DEFE komutunu kullanarak bir JNDI ad alanında saklayın.

Aşağıdaki komut sözdizimini kullanın:

```
DEFINE TYPE(name) [property]*
```

Yani, DEFINE yüklemi ve ardından *TYPE*(name) tarafından yönetilen bir nesne başvurusu ve ardından sıfır ya da daha fazla *özellik* (bkz. IBM WebSphere MQ classes for JMS nesnelerinin özellikleri başlıklı konuya bakın).

JMS nesnelerine ilişkin LDAP adlandırma konuları

Nesnenizi bir LDAP ortamında saklamak için, bazı kurallara uygun adlar vermelisiniz. Yönetim aracı, varsayılan bir önek ekleyerek adlandırma kurallarına uymanıza yardımcı olur.

Bir adlandırma kuralı, nesne ve alt bağlam adlarının cn= (ortak ad) ya da ou= (kuruluş birimi) gibi bir önek içermesi gerektiğini ifade eder.

Yönetim aracı, nesneye ve bağlam adlarına önek olmadan başvuruda bulunmanıza olanak sağlayarak LDAP hizmet sağlayıcılarının kullanımını basitleştirir. Bir önek belirtmezseniz, araç sizin sağladığınız ada otomatik olarak varsayılan bir önek ekler. LDAP için bu, cn=' dir.

JMSAdmin yapılandırma dosyasında NAME_PREFIX özelliğini “Listelenmeyen bir InitialContextÜreticisi 'ni WebSphere MQ JMS yönetim aracı ile kullanma” sayfa 898içinde açıklandığı gibi ayarlayarak varsayılan öneki değiştirebilirsiniz.

Bu, aşağıdaki örnekteki gösterilmektedir.

```
InitCtx> DEFINE Q(testQueue)

InitCtx> DISPLAY CTX

    Contents of InitCtx

      a  cn=testQueue              com.ibm.mq.jms.MQQueue

     1 Object(s)
       0 Context(s)
       1 Binding(s), 1 Administered
```

Sağlanan nesne adının (`testQueue`) bir öneki bulunmasa da, araç otomatik olarak LDAP adlandırma kuralına uygunluğu sağlamak için bir tane ekler. Benzer şekilde, `DISPLAY Q(testQueue)` komutunun da sunulması bu önekin de eklenmesine neden olur.

Java nesnelerini saklamak için LDAP sunucunuzun konfigürasyonunu tanımlamanız gerekebilir. Bu konfigürasyona yardımcı olacak bilgiler için LDAP sunucunuza ilişkin belgelere bakın.

JMS nesnesi yaratılırken örnek hata koşulları

Bir nesne oluşturduğunuzda, bir dizi ortak hata koşulu ortaya çıkabilir.

Aşağıda, bu hata koşullarıyla ilgili örnekler bulunmaktadır:

CipherSpec , CipherSuiteile eşlendi.

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) SSLCIPHERSUITE(RC4_MD5_US)
  WARNING: Converting CipherSpec RC4_MD5_US to
  CipherSuite SSL_RSA_WITH_RC4_128_MD5
```

Nesne için geçersiz özellik

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) PRIORITY(4)
  Unable to create a valid object, please check the parameters supplied
  Invalid property for a QCF: PRI
```

Özellik değeri için geçersiz tip

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) CCSID(english)
  Unable to create a valid object, please check the parameters supplied
  Invalid value for CCS property: English
```

Özellik çakışması-istemci/bindings

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) HOSTNAME(polaris.hursley.ibm.com)
  Unable to create a valid object, please check the parameters supplied
  Invalid property in this context: Client-bindings attribute clash
```

Özellik çakışması-Başlatma kullanıma hazırlama

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) SECEXITINIT(initStr)
  Unable to create a valid object, please check the parameters supplied
  Invalid property in this context: ExitInit string supplied
  without Exit string
```

Özellik değeri geçerli aralık dışında

```
InitCtx/cn=Trash> DEFINE Q(testQ) PRIORITY(12)
  Unable to create a valid object, please check the parameters supplied
  Invalid value for PRI property: 12
```

Bilinmeyen özellik

```
InitCtx/cn=Trash> DEFINE QCF(testQCF) PIZZA(ham and mushroom)
  Unable to create a valid object, please check the parameters supplied
  Unknown property: PIZZA
```

Aşağıda, bir JMS uygulamasından JNDI denetimli nesneler aranırken Windows üzerinde oluşabilecek hata koşulları örnekleri yer alıyor.

1. If you are using the WebSphere JNDI provider, com.ibm.websphere.naming.WsnInitialContextFactory, you must use a forward slash (/) to access administered objects defined in subcontexts; for example, jms/MyQueueName. Ters eğik çizgi (\) kullanırsanız, InvalidNamekural dışı durumu oluşur.
2. Sun JNDI sağlayıcısını kullanıyorsanız, com.sun.jndi.fscontext.RefFSContextFactory, alt bağlamlarda tanımlanan denetlenen nesnelere erişmek için ters eğik çizgi (\) kullanmanız gerekir; örneğin, ctx1\ \fred. Eğik çizgi (/) kullanırsanız, bir NameNotFoundException yayınlanır.

JMS yapılandırması için WebSphere MQ Explorer olanağının kullanılması

Use the IBM WebSphere MQ Explorer graphical user interface to create JMS objects from WebSphere MQ objects, and WebSphere MQ objects from JMS objects, as well as for administering and monitoring other WebSphere MQ objects.

Başlamadan önce

Before you create and configure JMS administered objects with the WebSphere MQ Explorer, add an initial context to define the root of the JNDI namespace in which the JMS objects are stored in the naming and directory service. Daha fazla bilgi için, JMS denetimli nesneler için IBM WebSphere MQ Explorer kullanıcı yardımına bakın.

Bu görev hakkında

You can perform the following tasks with the IBM WebSphere MQ Explorer, either contextually from an existing object in the IBM WebSphere MQ Explorer, or from within a create new object wizard. Bazı tipik görevler için WebSphere MQ Explorer kullanıcı yardımının örnekleri için WebSphere MQ Explorer yardımına bakın.

Yordam

- Aşağıdaki WebSphere MQ nesnelerinden herhangi birinden bir JMS Bağlantı Üreticisi yaratın:
  a) Yerel bilgisayarınızda ya da uzak bir sistemde bulunan bir WebSphere MQ kuyruk yöneticisi.
  b) Bir WebSphere MQ kanalı
  c) Bir WebSphere MQ dinleyicisi
- Add a WebSphere MQ queue manager to WebSphere MQ Explorer using a JMS Connection Factory
- WebSphere MQ kuyruğundan bir JMS kuyruğu yarat
- JMS kuyruğundan bir WebSphere MQ kuyruğu yarat
- Bir WebSphere MQ nesnesi ya da dinamik bir konu olabilen bir WebSphere MQ konusundan bir JMS konusu oluşturun
- Bir JMS konusundan bir WebSphere MQ konusu yaratın

WebSphere MQ Üstbilgileri paketinin kullanılması

WebSphere MQ Headers paketi, bir iletinin WebSphere MQ üstbilgilerini işlemek için kullanabileceğiniz yardımcı arabirimler ve sınıflar kümesi sağlar. Genellikle, komut sunucusunu kullanarak (Programlanabilir Komut Biçimi (PCF) iletileri kullanarak) denetim hizmetlerini gerçekleştirmek istediğinizde WebSphere MQ Üstbilgileri paketini kullanabilirsiniz.

Bu görev hakkında

WebSphere MQ Üstbilgileri paketi, `com.ibm.mq.headers` ve `com.ibm.mq.pcf` paketlerinde bulunur. Bu olanağı, WebSphere MQ tarafından Java uygulamalarında kullanılmak üzere sağlanan iki alternatif API için de kullanabilirsiniz:

- Java için WebSphere MQ sınıfları (WebSphere MQ Üstbilgileri Temel Java olarak da adlandırılır).
- Java Message Service için WebSphere MQ sınıfları (JMS için WebSphere MQ sınıfları, ayrıca WebSphere MQ JMS olarak da adlandırılır).

WebSphere MQ Base Java uygulamaları genellikle MQMessage nesnelerini işleyebilir ve Headers destek sınıfları WebSphere MQ Base Java arabirimlerini yerel olarak anladıkları için, bu nesnelerle doğrudan etkileşimde bulunabilir.

WebSphere MQ JMS olanağında, iletiye ilişkin bilgi yükü tipik olarak bir Dize ya da bayt dizisi nesnesidir ve DataInput ve DataOutput akımıyla işlenebilir. WebSphere MQ Üstbilgiler paketi, bu veri akışlarıyla etkileşimde bulunmak için kullanılabilir ve WebSphere MQ JMS uygulamaları tarafından gönderilen ve alınan MQ iletilerini işleme almak için uygundur.

Therefore, although the WebSphere MQ Headers package contains references to the WebSphere MQ Base Java package, it is also intended for use within WebSphere MQ JMS applications and is suitable for use within Java Platform, Enterprise Edition (Java EE) environments.

WebSphere MQ Headers paketini kullanabileceğiniz tipik bir yöntem, yönetim iletilerini PCF (Programmable Command Format; Programlanır Komut Biçimi) içinde (örneğin, aşağıdaki nedenlerden biri için) değiştirmenize yardımcı olur:

- Bir WebSphere MQ kaynağıyla ilgili ayrıntılara erişmek için.
- Bir kuyruğun derinliğini izlemek için.
- Bir kuyruğa erişimi engellemek için.

By using PCF messages with the WebSphere MQ JMS API, this kind of administration of application-centric resources can be performed from within Java EE applications without having to resort to using the WebSphere MQ Base Java API.

Yordam

- Java için WebSphere MQ sınıflarına ilişkin ileti üstbilgilerini işlemek üzere WebSphere MQ Üstbilgileri paketini kullanmak için “Java için WebSphere MQ sınıflarıyla kullanma” sayfa 905konusuna bakın.
- JMS için ileti üstbilgilerini işlemek üzere WebSphere MQ Üstbilgileri paketini kullanmak için “JMS için WebSphere MQ sınıflarıyla kullanma” sayfa 906konusuna bakın.

Java için WebSphere MQ sınıflarıyla kullanma

Java uygulamaları için WebSphere MQ sınıfları tipik olarak MQMessage nesnelerini yönlendirir ve Java arabirimleri için WebSphere MQ sınıflarını yerel olarak anladıkları için, Üstbilgiler destek sınıfları bu nesnelerle doğrudan etkileşime girebilirler.

Bu görev hakkında

WebSphere MQ provides some sample applications that demonstrate how to use the WebSphere MQ Headers package with the WebSphere MQ Base Java API (WebSphere MQ classes for Java).

Örneklerde iki şey var:

- Bir denetim işlemini gerçekleştirmek ve yanıt iletisini ayrıştırmak için bir PCF iletisinin nasıl yaratılacağı.
- Java için WebSphere MQ sınıflarını kullanarak bu PCF iletisinin nasıl gönderileceği.

Kullandığınız platforma bağlı olarak, bu örnekler, WebSphere MQ kurulumunuzun `samples` ya da `tools` dizininde bulunan `pcf` dizini altında kurulur (bkz. “Java için WebSphere MQ sınıflarına ilişkin kuruluş dizinleri” sayfa 630).

Yordam

1. Bir denetim işlemi gerçekleştirmek için bir PCF iletisi yaratın ve yanıt iletisini ayrıştıraın.
2. Java için WebSphere MQ sınıflarını kullanarak bu PCF iletisini gönderin.

İlgili kavramlar

“Java için WebSphere MQ sınıflarıyla WebSphere MQ ileti üstbilgileri ele alma” sayfa 651
Java sınıfları farklı tipte ileti üstbilgisi gösteririr. İki yardımcı sınıfı da sağlanır.

“Java için WebSphere MQ sınıflarıyla PCF iletilerinin işlenmesi” sayfa 656
Java sınıfları PCF tarafından yapılandırılmış iletiler yaratmak ve ayrıştırmak ve PCF isteklerini göndermek ve PCF yanıtlarını toplamak için sağlanmıştır.

JMS için WebSphere MQ sınıflarıyla kullanma

To use the WebSphere MQ Headers with the WebSphere MQ classes for JMS, you carry out the same essential steps as for WebSphere MQ classes for Java . The PCF message can be created and the response parsed in exactly the same way by using the WebSphere MQ Headers package and the same sample code as for WebSphere MQ classes for Java .

Bu görev hakkında

WebSphere MQ API 'yı kullanarak bir PCF iletisi göndermek için, ileti bilgi yükü bir JMS Bytes İletisine yazılmalı ve standart JMS API' ları kullanılarak gönderilmelidir. Tek dikkat edilmesi gereken, iletinin bir JMS RFH2 ya da MQMD ' deki belirli değerleri içeren başka bir üstbilgileri içermemesi gerekir.

Bir PCF iletisi göndermek için aşağıdaki adımları tamamlayın. PCF iletisinin yaratıldığı yol ve bilgiler yanıt iletisinden çıkarıldığında, Java için WebSphere MQ sınıfları için aynıdır (bkz. “Java için WebSphere MQ sınıflarıyla kullanma” sayfa 905).

Yordam

1. SYSTEM.ADMIN.COMMAND.QUEUE.

   WebSphere MQ JMS uygulamaları, PCF iletilerini SYSTEM.ADMIN.COMMAND.QUEUE(Kuyruk) ve bu kuyruğu gösteren bir JMS Hedefi (JMS Hedef) nesnesine erişmeniz gerekir. Hedef, aşağıdaki özellikler kümesine sahip olmalıdır:

   ```
   WMQ_MQMD_WRITE_ENABLED = YES
   WMQ_MESSAGE_BODY = MQ
   ```

   WebSphere Application Server kullanıyorsanız, Hedefte özel özellikler olarak bu özellikleri tanımlamanız gerekir.

   Hedef programsal olarak bir uygulama içinden yaratmak için aşağıdaki kodu kullanın:

   ```
   Queue q1 = session.createQueue("SYSTEM.ADMIN.COMMAND.QUEUE");
   ((MQQueue) q1).setIntProperty(WMQConstants.WMQ_MESSAGE_BODY,
       WMQConstants.WMQ_MESSAGE_BODY_MQ);
   ((MQQueue) q1).setMQMDWriteEnabled(true);
   ```

2. Bir PCF iletisini doğru MQMD değerlerini içeren bir JMS Byte iletisine dönüştürün.

   Bir JMS Byte iletisi yaratılmalıdır ve PCF iletisi yazılmalıdır. Bir yanıt kuyruğu yaratılması gerekiyor, ancak bunun belirli bir ayarı olmaması gerekiyor.

   Aşağıdaki örnek kod parçacığı, JMS Byte İletisi 'nin nasıl yaratılacağı ve bir com.ibm.mq.headers,pcf.PCFMessage nesnesi nasıl yazılacağı gösterilmektedir. PCFMessage nesnesi (pcfCmd) önceden WebSphere MQ Üstbilgileri paketi kullanılarak oluşturulmuş. (PCFMessage 'ı yüklemek için paketin com.ibm.mq.headers.pcf.PCFMessageolduğunu unutmayın).

   ```
    // create the JMS Bytes Message
   final BytesMessage msg = session.createBytesMessage();

   // Create the wrapping streams to put the bytes into the message payload
   ByteArrayOutputStream baos = new ByteArrayOutputStream();
   DataOutput dataOutput = new DataOutputStream(baos);

   // Set the JMSReplyTo so the answer comes back
   msg.setJMSReplyTo(new MQQueue("adminResp"));

   // write the pcf into the stream
   pcfCmd.write(dataOutput);
   baos.flush();
   msg.writeBytes(baos.toByteArray());

   // we have taken control of the MD, so need to set all
   ```

```
// flags in the MD that we require - main one is the format
msg.setJMSPriority(4);
msg.setIntProperty(WMQConstants.JMS_IBM_MQMD_PERSISTENCE,
  CMQC.MQPER_NOT_PERSISTENT);
msg.setIntProperty(WMQConstants.JMS_IBM_MQMD_EXPIRY, 300);
msg.setIntProperty(WMQConstants.JMS_IBM_MQMD_REPORT,
    CMQC.MQRO_PASS_CORREL_ID);
msg.setStringProperty(WMQConstants.JMS_IBM_MQMD_FORMAT, "MQADMIN");

// and send the message
sender.send(msg);
```

3. İletiyi gönderin ve standart JMS API ' larını kullanarak yanıtı alın.
4. Yanıt iletisini işlenmek üzere bir PCF iletisine dönüştürün.

   Yanıt iletisini almak ve bunu bir PCF iletisi olarak işlemek için aşağıdaki kodu kullanın:

```
 // Get the message back
BytesMessage msg = (BytesMessage) consumer.receive();

// get the size of the bytes message & read into an array
int bodySize = (int) msg.getBodyLength();
byte[] data = new byte[bodySize];
msg.readBytes(data);

// Read into Stream and DataInput Stream
ByteArrayInputStream bais = new ByteArrayInputStream(data);
DataInput dataInput = new DataInputStream(bais);

// Pass to PCF Message to process
PCFMessage response = new PCFMessage(dataInput);
```

İlgili kavramlar

“JMS iletileri” sayfa 775
JMS iletileri bir üstbilgiden, özelliklerden ve bir gövden oluşur. JMS, ileti gövdesinin beş tipini tanımlar.

WebSphere MQ' da Web hizmetlerini kullanma

SOAP için IBM WebSphere MQ iletimi ya da HTTP için IBM WebSphere MQ köprüsü kullanılarak Web hizmetleri için IBM WebSphere MQ uygulamaları geliştirebilirsiniz.

SOAP için IBM WebSphere MQ iletimi SOAP için bir JMS iletimi sağlar. SOAP için IBM WebSphere MQ iletimi de, Microsoft Windows Communication Foundation, WebSphere Application Server ve CICS Transaction Server gibi diğer ortamlara da tümleştirilmiştir.

SOAP için IBM WebSphere MQ iletimi ile ilgili daha fazla bilgi için bkz. “SOAP içinWebSphere MQ iletimi” sayfa 908.

With IBM WebSphere MQ bridge for HTTP, client applications can exchange messages with IBM WebSphere MQ without the need to install a WebSphere MQ MQI client. You can call WebSphere MQ from any platform or language with HTTP capabilities.

HTTP için IBM WebSphere MQ köprüsü ile ilgili daha fazla bilgi için bkz. “HTTP içinWebSphere MQ köprüsü” sayfa 982.

İlgili kavramlar

“Uygulama geliştirme kavramları” sayfa 7
IBM WebSphere MQ uygulamalarını yazmak için yordamsal ya da nesne yönelimli dil seçenekleri kullanabilirsiniz. Uygulama geliştiriciler için yararlı olan IBM WebSphere MQ kavramlarına ilişkin bilgiler için bu konudaki bağlantıları kullanın.

“Kullanılacak programlama diline karar verme” sayfa 75
Bu bilgileri, IBM WebSphere MQtarafından desteklenen programlama dilleri ve çerçeveleri ve bu dilleri kullanırken dikkat edilmesi gereken bazı noktalar hakkında bilgi edinmek için kullanın.

“IBM WebSphere MQ uygulamalarını tasarlama” sayfa 86
Uygulamalarınızın kullanımınıza sunulan platformlardan ve ortamlardan nasıl yararlanabileceğinize karar verdiğinizde, WebSphere MQtarafından sunulan özelliklerin nasıl kullanılacağına karar vermeniz gerekir.

“Örnek WebSphere MQ programları” sayfa 92

Farklı platformlardaki örnek WebSphere MQ programları hakkında bilgi edinmek için bu konu toplamasını kullanın.

"Kuyruğa alma uygulaması yazılıyor" sayfa 186
Kuyruk yöneticisi, yayınlama/abone olma, nesneleri açma ve kapama nesnelerine bağlanma ve bağlantıyı kesme, kuyruğa alma ve bağlantı kesme hakkında bilgi edinmek için bu bilgileri kullanın.

"İstemci uygulamaları yazılıyor" sayfa 335
What you need to know to write client applications on WebSphere MQ.

"Yayınlama/abone olma uygulamaları yazılıyor" sayfa 266
Yayınlama/abone olma WebSphere MQ uygulamalarını yazmaya başlayın.

"IBM WebSphere MQ uygulaması oluşturulması" sayfa 409
Farklı platformlarda bir IBM WebSphere MQ uygulaması oluşturma hakkında bilgi edinmek için bu bilgileri kullanın.

"Program hatalarının işlenmesi" sayfa 526
Bu bilgiler, bir çağrı yaparken ya da iletisi son hedefine teslim edildiğinde, uygulama MQI çağrılarınızla ilişkili hataları açıklar.

SOAP içinWebSphere MQ iletimi

SOAP için WebSphere MQ iletimi SOAP için bir JMS iletimi sağlar. SOAP için WebSphere MQ iletimi aynı zamanda Microsoft Windows Communication Foundation, WebSphere Application Server ve CICS Transaction Server gibi diğer ortamlara da tümleştirilir.

SOAP için IBM WebSphere MQ iletine giriş

SOAP için IBM WebSphere MQ iletimi SOAP için bir JMS iletimi sağlar. WebSphere MQ SOAP göndericisi ve dinleyici, Web hizmetlerini aramanız için bir araç sağlar.

WebSphere MQ SOAP dinleyicisi, .NET Framework 1, .NET Framework 2 ve Axis 1.4tarafından barındırılan hizmetleri destekler. WebSphere MQ SOAP göndericisi, .NET Framework 1, .NET Framework 2, Axis 1.4 ve Axis2üzerinde çalışan web hizmetleri istemcilerini destekler. İstemciler, bir WebSphere MQ sunucusu ya da istemci uygulaması olabilir. SOAP için IBM WebSphere MQ iletimi de, Microsoft Windows Communication Foundation, WebSphere Application Server ve CICS Transaction Server gibi diğer ortamlara da tümleştirilmiştir.

Microsoft Windows Communication Foundation bütünleştirmesi, .NET Framework 3 için IBM WebSphere MQ desteğinin bir parçasıdır.

SOAP için IBM WebSphere MQ iletimi, SOAP iletilerini IBM WebSphere MQüzerinden JMS kullanarak aktarmak için bir protokol ve araçlar kümesidir. Çizelge 137 sayfa 908içinde gösterildiği gibi farklı uygulama ortamları için farklı şekillerde paketlenmiştir.

Çizelge 137. SOAP uygulama ortamları içinIBM WebSphere MQ iletimi

	Ek WebSphere MQ bileşenleriyle tümleştirilmiştir	**Bir çerçeveye tümleştirilmiştir**
WebSphere MQ kuruluşunun bir parçası olarak sağlanır.	.NET Framework 1 .NET Çerçeve 2 Ekseni 1.4	Windows Communication Foundation (.NET Framework 3) Axis2 (yalnızca istemci)
Başka bir yazılım paketinde sağlanan		WebSphere Application Server CICS Transaction Server 4.1 WebSphere ESB WebSphere Process Server for Multiplatforms

SOAP için IBM WebSphere MQ iletimi bir uygulama çerçevesinin bütünleştirilmesi, web hizmetlerinin IBM WebSphere MQ' e geliştirilmesini ve konuşlandırılmasını kolaylaştırır.

Ek IBM WebSphere MQ SOAP bileşenleriyle, hizmetleri geliştirmek ve konuşlandırmak için doğrudan WebSphere MQ SOAP bileşenleriyle etkileşimde bulunabilirsiniz. Web hizmetlerini ve web hizmeti istemcilerini IBM WebSphere MQ' e yapılandırmak ve konuşlandırmak için IBM WebSphere MQ SOAP araçlarını kullanın.

Tümleşik ortamlarda, geliştirme ve devreye alma daha basittir. Bir SOAP HTTP web hizmeti geliştirirken ve konuşlandırdığınızda, geliştirme ve devreye alma için aynı araçları kullanıyorsunuz. You must still configure the IBM WebSphere MQ queues, channels, and queue managers that you require using WebSphere MQ tools.

IBM WebSphere MQ SOAP istemcilerini ve sunucularını bu ortamlardan herhangi birinden karışık olarak eşleştirebilir ve eşleştirebilirsiniz.

Yararlar

SOAP için WebSphere MQ iletimi var olan IBM WebSphere MQ kullanıcılarına şu birincil kullanıcı yararları sunar:

Var olan web hizmetlerini bağlamak için IBM WebSphere MQ ağınızı kullanma.

Hizmetler, konuşlandırdığınız diğer paketlenmiş yazılım uygulamalarına arabirim olarak sağlanan, sizin yazdığınız ya da hizmetlerinizin olduğu hizmetler olabilir.

Avantaj, web hizmetlerini bağlamak için var olan WebSphere MQ ağınızı kullanmaktan gelir. IBM WebSphere MQ iletimi, yönetilen ve güvenilir bir kuyruğa alınan ileti sistemi hizmeti olmanın avantajına sahiptir.

Writing new applications, or converting existing applications, to use SOAP rather than IBM WebSphere MQ interfaces.

Genellikle uygulamalar, başka bir uygulamayla bütünleştirmek için belirli bir WebSphere MQ bağdaştırıcısının geliştirilmesini gerektirir. Bağdaştırıcıların iki bölümü vardır: bağlaç parçası, iletimden ve iletiden ileti alan ve alan ve uygulamaya özgü biçimlere veri dönüştüren bağdaştırıcı parçası. Her bir uygulama çiftinin bütünleştirilmesi yeni bir sorundur.

SOAP 'ın yararı, uygulama arabirimlerini tanımlamak için SOAP' ta standartlaştırılıyor ve daha sonra bir iletilme seçeneği ortaya çıkıyor. Uygulamaya özgü bağdaştırıcılar yazmanıza gerek yoktur ve bağlayıcı olarak IBM WebSphere MQ ya da HTTP ' yi kullanmayı seçebilirsiniz. Hangi taşıma işlemi için gerekli hizmet ve bağlanırlık nitelerine bağlı olarak seçim yapabilirsiniz.

For existing SOAP over HTTP users, the benefit of WebSphere MQ transport for SOAP comes from using a managed and reliable asynchronous transport. Avantajların iki şekli var.

Kullanılabilirlik ve performans için gerçek bir zamanuyumsuz programlama modeli.
Bir zamanuyumsuz istemci arabirimi kullanarak, istemci ve hizmet uygulamalarının aynı anda kullanılabilir olması gerekmez. İstemci tarafından gönderilen istekler, hizmet verilinceye kadar saklanacaktır.

Güvenilir ve kullanılabilir olmak üzere tasarlanmış, hazır oluşturulmuş bir yönetilen ağ.

İletim olarak IBM WebSphere MQ ' ı seçerek, güvenilir ileti sistemi sağlayan bir yönetilen ağ kullanmanın avantajını elde etmiyorsunuz.

Buna karşılık, TCP/IP üzerinden HTTP ve FTP gibi aktarma yönetilmez. Yönetilmeyen bir ağ, tahmin edilemeyen bağlantılar için idealdir: Daha az yönetim görevi vardır.

Özet

SOAP içinIBM WebSphere MQ iletimi aşağıdaki bileşenleri sağlar:

- SOAP/JMS iletimi bağ tanımı, bir SOAP hizmetini JMS iletiminde bağlamak için WSDL belgelerinde kullanılır. SOAP/JMS bağ tanımının WebSphere MQ somutlaması, iki biçimden birini alan bir URI kullanır:

SOAP içinWebSphere MQ iletimi

```
jms:/queue?&Name=Value&Name=Value...
```

W3C aday önerisi içinWebSphere MQ aktarım kanalı biçimi

```
jms:queue:qName?connectionFactory=connectQueueManager(qMgrName)&Name=Value&Name=Value...
```

- Bir SOAP iletisinin bir WebSphere MQ iletisine eşlenmesine neden olur.
- SOAP isteklerini almak için iki IBM WebSphere MQ SOAP dinleyicisi, biri Java için, diğeri de .NET Framework 1 ya da .NET Framework 2 için. Dinleyiciler SOAP isteğini işlemek için .NET ya da Axis 1.4 kullanır.
- Two IBM WebSphere MQ SOAP senders to create IBM WebSphere MQ SOAP requests. Web hizmetleri istemcileri, `jms:` SOAP isteklerini işlemek için bir göndericiyle kaydolur.
- SOAP iletileri için WebSphere MQ Transport 'u göndermek ve almak için Windows Communication Foundation (WCF) ile (bazen .NET 3 olarak da bilinir) bütünleştirme.
- Integration of the client with Axis2, sometimes known as JAX-WS, to send WebSphere MQ Transport for SOAP or W3C SOAP JMS messages.
- SOAPkomutunu, SOAP için IBM WebSphere MQ iletimi kullanarak bir web hizmeti konuşlandırmak için geliştirme ve çalıştırma zamanı bileşenleri ve komut dosyaları oluşturan **amqwdeployWMQService**komutu.
- Örnek Java ve .NET istemcisi ve hizmet kodu.
- Sınıf yolunu ve diğer yardımcı program komut dosyalarını ayarlamak için kullanılan komut dosyası.

Tümleşik ortamlarda, gönderen ve dinleyici, geliştirme ve konuşlandırma araçlarına ilişkin uzantılar olarak her bir ortama tümleştirilmiştir.

SOAP ve WebSphere MQ' nun bütünleştirilmesi

SOAP için WebSphere MQ iletimi SOAP ve Web hizmetleri araçlarını ve çalıştırma zamanını, WebSphere MQ ile SOAP için HTTP ' ye alternatif bir iletim olarak sağlar. Bir taşıma olarak SOAP için WebSphere MQ iletimi kullanabilmek için var olan Web hizmetlerini değiştirmeniz gerekmez. İletim, SOAP/JMS için özel bir URI biçimi kullanır. SOAP/JMS için W3C URI biçimi, Axis2 istemcilerinin sınırlı bir şekilde desteklenir.

.NET Framework 1, .NET Framework 2 ve Axis 1.4 ortamlarındaki istemcilere ek bir kod satırı eklenmelidir. Axis 2 ve Windows Communication Foundation (WCF) istemcilerinde ek kod gerekli değildir. WebSphere MQ SOAP dinleyicisi, .NET Framework 1, .NET Framework 2 ve Axis 1.4 ortamlarındaki hizmetleri çalıştırır. SOAP için WebSphere MQ iletimi, WCF, CICS ve WebSphere Application Serverde içinde olmak üzere diğer bazı uygulama sunucusu ortamlarında tümleştirilir.

SOAP nedir?

SOAP[9]Uygulamaların istekleri, yanıtları ve datagramları değiş tokuş etmek için kullandıkları iletilerin ve etkileşim protokollerinin standartlaştırılmış biçimini açıklar. SOAP, iletileri aktarmak için kullanılan iletim ortamından ve iletileri gönderen ve alan uygulama ortamının bağımsız bir uygulamasıdır. W3C , SOAP Sürüm 1.2 ' u kısa bir şekilde tanımlıyor:

SOAP Sürüm 1.2 , dağıtılmış ve dağıtılmış bir ortamdaki eşler arasında yapılandırılmış ve tip atanmış bilgi alışverişi için kullanılabilecek XML tabanlı bilgilerin tanımlamasını sağlar.[10].

SOAP ' ı kullanmak için, HTTP, e-posta ya da WebSphere MQgibi bir aktarıma bağlı olmalıdır.

SOAP protokolü bağ tanımı çerçevesi, HTTP gibi başka bir protokolün üzerine bir SOAP iletisi taşımaya ilişkin kurallar kümesidir. SOAP Sürüm 1.2 Bölüm 2: Adresler (Second Edition) , SOAP HTTP bağ tanımını açıklar.

[9] Tarihsel olarak kısaltma, Basit Nesne Erişimi Protokolü için durdu.
[10] W3C: SOAP Sürüm 1.2 Part 0

The W3C candidate recommendation, 4 June 2009, SOAP over Java Message Service 1.0, describes the recommendation for the SOAP JMS binding. JMS bir iletim protokolü değil, bir API belirtimi olduğu için, JMS SOAP önerisi SOAP JMS iletilerinin aktarım kanalını açıklamaz. Bu, SOAP etkileşimi protokollerini ve JMS API bağ tanımını açıklar. Sonuç olarak, JMS SOAP önerisini kullanırken, SOAP istemcisi ve SOAP sunucusu için aynı JMS somutlamasını kullanmaya devam etmelisiniz. JMS ' nin herhangi bir somutlaması üzerinde bir SOAP JMS uygulamasının çalıştırılabilmesini sağlar. Bir JMS uygulaması, hem sunucu, hem de JMS uygulaması JCA belirtimiyle uyumlu olursa, bir J2EE uygulama sunucusuna takılabilir. WebSphere MQ JMS, JCA belirtimine uygundur ve uyumlu bir uygulama sunucusuna takılabilir.

WebSphere MQ for SOAP bağ tanımı, önerilen W3C standardı gibidir, ancak aynı değildir. Kullanımı MQRFH2 SOAP ayarlarıbaşlıklı konuda açıklanmaktadır. Unlike the W3C candidate recommendation, the SOAP binding is not formally specified. Effectively, it is the HTTP binding, and the service address takes the form, `jms:/queue?name=value&name=value...`, rather than `http://authority/path?query#fragment`. `jms:` , resmi olarak kayıtlı bir IANA URI 'si şeması değildir.

Web hizmeti nedir?

SOAP, farklı platformlarda çalışan programların farklı platformlarda çalıştırılmasını ve çeşitli taşıma protokollerini kullanarak iletişim kurmasını sağlar. SOAP, protokol belirtimidir. Web hizmeti, Internet iletişim kuralları kullanılarak erişilebilen bir SOAP arabirimiyle hizmet sağlayan bir uygulamadır.

SOAP ' ın önemli bir amacı, müşterilerin kolayca kullanabilecekleri hizmetleri sağlamaktır. Bir hizmeti kullanmak üzere bir istemci tasarladığınızda, dış belgelere başvurmadan hizmeti çağırmak için çağrıyı programlayabilirsiniz. Hizmet arabirimleri, bir WSDL belgesinde XML ' de açıklanmıştır. The query, `http://authority/path?wsdl`, returns the WSDL description of a SOAP service.

İpucu: WebSphere MQ'yı kullanmak için bir Web hizmeti konuşlandırdığınızda, standart WSDL sorgularının işlev görebilmesi için hizmeti HTTP' ye konuşlandırın.

Web hizmetleri geliştirilmesi

Web hizmetlerinin bir istemcisi ve bir hizmet bölümü vardır. Hizmet, WSDL ' deki arabirim açıklamasından başlayarak ya da hizmet sınıfını yazmak için kurallara uyarak ilk önce yazılır. Web hizmeti araç takımları, bir sınıfın arabirim tanımlamasından WSDL oluşturmak için yardımcı programlara sahiptir; örneğin, **`java2wsdl`** ya da **`disco`**. Ayrıca, WSDL arabirimi tanımlamalarından kaynak iskeletleri oluşturmak ya da bunları sınıflamak için araçlar da vardır; örneğin, **`wsdl2java`**, **`wsimport`**ya da **`wsdl`**. Eski, en alt gelişim olarak bilinir ve ikincisi de aşağı yukarı aşağı inmektedir.

SOAP ' a ilişkin WebSphere MQ iletiminde **`amqwdeployWMQService`** komutu, WSDL, istemci sınırlı kod öbekleri ve istemci yetkili sunucuları oluşturmak için bu araçları kullanır.

Web hizmetleri tipik olarak, belirli bir uygulama sunucusu ortamında hedeflenen tümleşik bir geliştirme ortamı kullanılarak yazılır:

Eclipse IDE for Java EE Developers
Axis 2 için Web hizmetleri yaratır. JAX-RPC ve JAX-WS ' i destekler

Rational Application Developer V7.5
WebSphere Application Server V7 ve önceki sürümler için ve Axis için de Web hizmetleri yaratır. JAX-RPC ve JAX-WS ' ı destekler.

WebSphere Integration Developer V6.2
WebSphere Process Server ve WebSphere ESB için Web hizmetleri yaratır. JAX-RPC ve JAX-WS ' ı destekler.

Visual Studio 2008 (Sürüm 9)
.NET Framework 3.5 ve önceki yayın düzeylerine ilişkin Web hizmetleri yaratır (Windows Communication Foundation)

Visual Studio 2005 (Sürüm 8)
.NET Framework 2 ve öncesi için Web hizmetleri yaratır

SOAP için WebSphere MQ iletimi ile birlikte bu araçlardan herhangi birini kullanabilirsiniz. Once you have developed a service to use with HTTP, use the **`amqwdeployWMQService`** tool to deploy the services to

use WebSphere MQ as a transport. Araçtan çıktıyı kullanarak yeni bir istemci yazabilir ya da var olan istemcilerinizi SOAP için WebSphere MQ iletimi kullanacak şekilde değiştirebilirsiniz.

SOAP için WebSphere MQ iletimi uygulama ortamına bütünleştirildiyse, **amqwdeployWMQService** aracını kullanmanız ya da istemci kodunu değiştirmeniz gerekmez. The client SOAP layer directs client requests that have a URI with the prefix `jms:` to the WebSphere MQ transport for SOAP. Sunucu SOAP katmanı, SOAP for `jms:` SOAP isteklerini beklemek için SOAP için WebSphere MQ iletimi çağırır ve SOAP için WebSphere MQ iletimi yanıtlarını döndürür.

Tipik olarak, .NET hizmetleri, WSDL arabirimi tanımlamaları kullanılarak, kodda Web hizmeti ek açıklamaları ve Java hizmetleri üst kısmında Web hizmeti ek açıklamaları kullanılarak alt düzey olarak geliştirilmiştir. Java Standard Edition Sürüm 6, JAX-WS 2.0' ı destekliyorsa ve hizmet arabirimlerinin tanımını nitelemek için ek açıklamalar kullandıkça, yaklaşımlardaki fark daraltılır. Aşağıda en çok Java hizmetlerini en üst düzey aşağı doğru geliştirmek kadar kolay bir iş. Hangi yaklaşımı seçiniz, bir geliştirme yöntemi meselesidir.

Web hizmetleri istemcisi, hizmet sonrasında, WSDL hizmeti tanımlaması ve oluşturulan istemci sınırlı kod öbekleri ve yetkili sunucular kullanılarak yazılır. Bazı uygulamalarda, hizmet tanımı, istemci yazıldığı zaman bilinmez. İstemci hizmet WSDL ' yi alır ve devingen olarak hizmet istekleri yaratır. Hizmet tanımının daha yaygın olarak bilinmesi, ancak hizmetin devreye alındığı adres olarak bilinmektedir. Web hizmeti araç takımı, istemci için hizmet istekleri yapmak üzere kullanılacak arabirimleri oluşturur. İstemci, gereken hizmet adresini sağlar. Üçüncü durumda, WSDL bir istemci gereksinmesi için gereken tüm bilgileri içerir. WSDL hem arabirimi, hem de hizmetin adresini içerir. Web hizmeti araç takımı tarafından oluşturulan kod, bir hizmetten istekte bulunmaya ilişkin tüm bilgileri istemci tarafından gerekli kıldı.

SOAP için WebSphere MQ iletimi ile bu üç stilden herhangi birini kullanabilirsiniz.

Web hizmeti uygulama ortamları

Web hizmeti araç takımları, bir hizmetin WSDL tanımlamasından, SOAP isteklerinde ve yanıtlarında aktarılan bayt akışlarına ilişkin bir eşleme gerektirir. Bayt akışı, SOAP belirtimiyle tanımlanır ve SOAP zarfında bulunur. SOAP zarfı Şekil 166 sayfa 912içinde gösterilir.

```
<?xml version='1.0'?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Header> <!-- optional -->
<!-- headers... -->
</soap:Header>
<soap:Body>
<!-- payload or fault message -->
</soap:Body>
</soap:Envelope>
```

Şekil 166. SOAP zarfı

SOAP zarfı ile dil bağ tanımı ve geri arasındaki eşleme, parça standartlaştırılır ve kısmen özel olur. Eşleme, .NET mimarisine temel olur ve CLR ' nin (Common Language Runtime; Ortak Dil Çalıştırma Zamanı) bir parçası olarak sağlanır. Eşleme, JAX belirtimlerine göre Java 'da standartlaştırılır. Java eşlemeleri standartlaştırıldığından, Java Web hizmeti istemcileri ve hizmetleri Java tabanlı farklı uygulama ortamları arasında taşınabilir. JAX-RPC (bazen JAX-WS 1.0olarak adlandırılır), bugün kullanılan en çok kullanılan eşlemedir. Bu, Axis 1.4tarafından desteklenir. JAX-WS (bazen JAX-WS 2.0olarak adlandırılır), büyük ölçüde geliştirilmiş bir standarttır ve hızla JAX-RPC ' nin yerini alır. JAX-WS, 2.0ekseni tarafından desteklenir. WebSphere MQ 7.0.1 , JAX-WS ve Axis 2 özelliğini desteklemiyor.

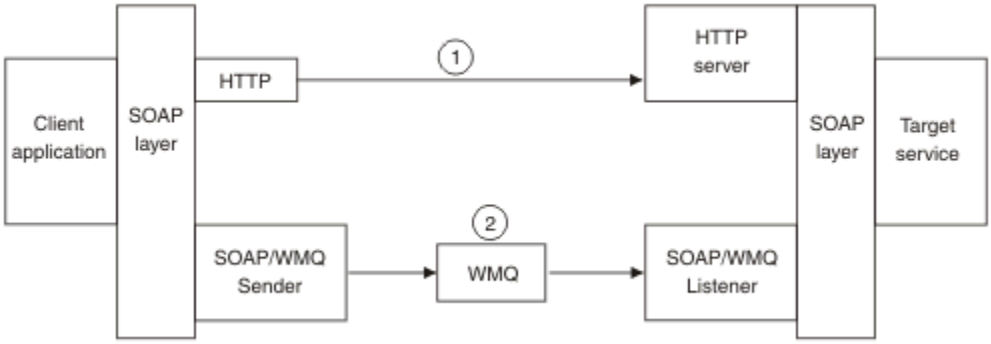
WebSphere MQ SOAP for SOAP, SOAP zarfının içeriğini değiştirmez ve içerikler iletim işlemini etkilemez. Dil bağlamaları, SOAP için WebSphere MQ iletimi işlemlerini etkiler. WebSphere MQ 7.0.1 , SOAP için WebSphere MQ iletimi ile birlikte verilen kod ve yardımcı programları kullanarak .NET Framework 1, .NET Framework 2 ve Axis 1.4 ürünlerini destekler. .NET Framework 3 ve 3.5 içinde SOAP için WebSphere iletimi desteği, Windows Communication Foundation için WebSphere MQ özel kanalı kullanılarak gerçekleştirilir.

Diğer SOAP geliştirme ve çalıştırma zamanı ortamları, SOAP için WebSphere MQ iletimi için destek verebilir ve farklı dilleri destekleyebilir. Örneğin, CICS üzerinde çalışan Web hizmetleri COBOL ve PL/1gibi dilleri destekler.

Not: Kullanılan eşleme, Web hizmetlerinin birlikte çalışabilirliği için hiçbir fark vermez. .NET, JAX-RPC ve JAX-WS eşlemelerini kullanarak, istemcileri ve hizmetleri karışık olarak eşleştirebilir ve eşleştirebilirsiniz.

SOAP için WebSphere MQ iletimi nedir?

SOAP içinWebSphere MQ iletimi, bir SOAP bağ tanımı ve bir Web hizmetleri araç takısıdır. Bunlar, uygulamaların HTTP iletilerini HTTP yerine WebSphere MQ kullanarak SOAP iletilerini değiş tokuş etmeye olanak sağlar. Şekil 167 sayfa 913 SOAP iletimi olarak HTTP 'ye alternatif olarak WebSphere MQ ' u gösterir.

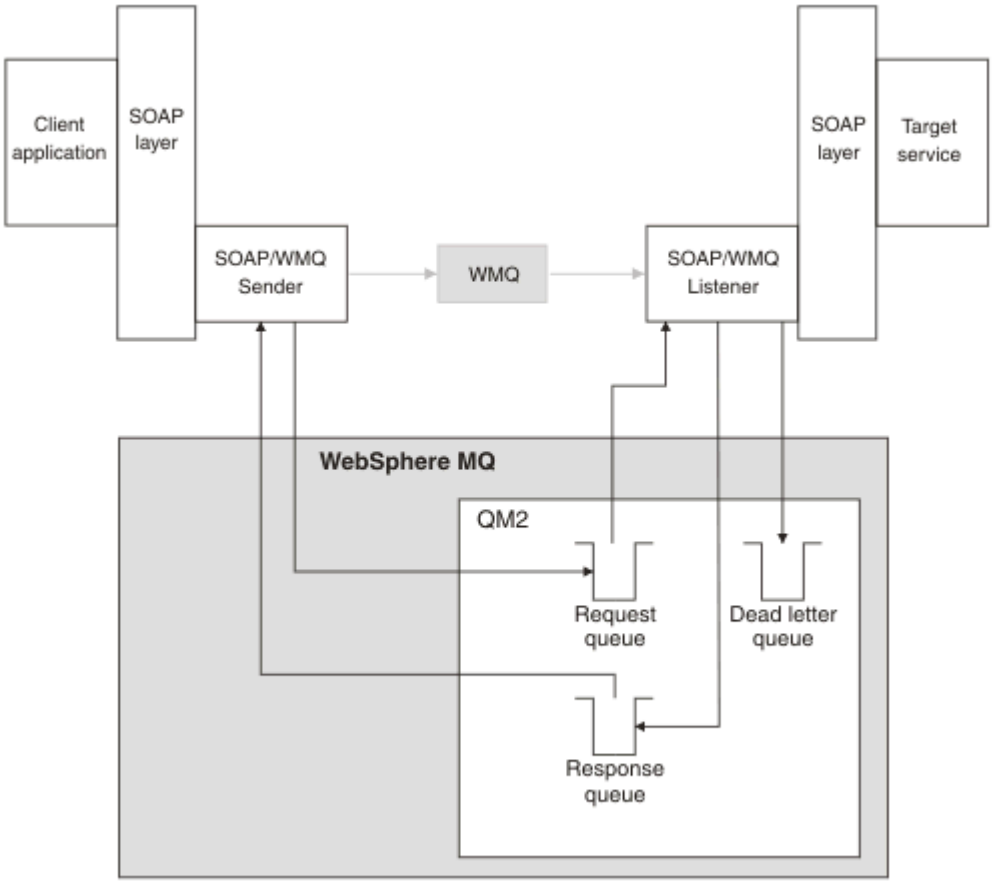


Şekil 167. SOAP için WebSphere MQ iletimi-genel bakış

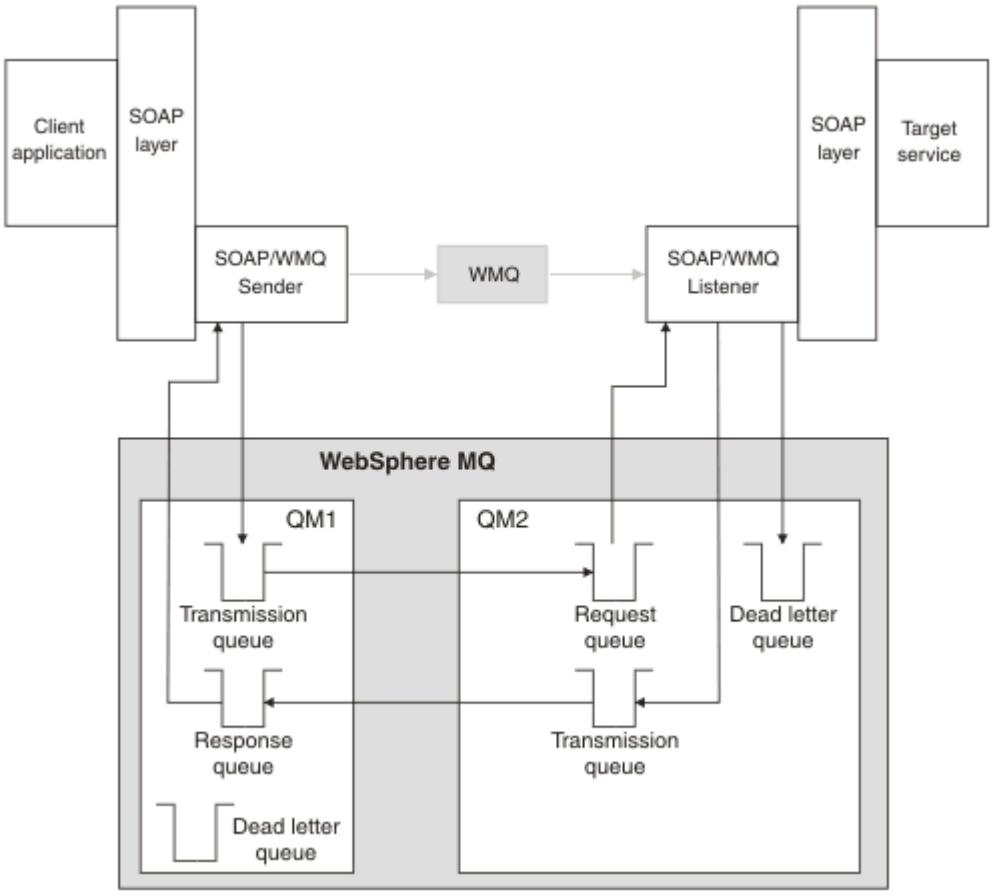
HTTP üzerinden SOAP, çizgede (1) olarak gösterilir. İstemci SOAP katmanı bir isteği SOAP iletisine dönüştürür ve HTTP bileşeni TCP/IP üzerinden gönderir. HTTP sunucusu bileşeni, genellikle 80 numaralı TCP/IP kapısındaki HTTP isteklerini dinler. İstek bir SOAP hizmetiyorsa, HTTP sunucusu bileşeni SOAP katmanını yöntem çağrısına dönüştürmek için SOAP katmanını çağırır. Daha sonra yanıtı döndürür.

SOAP over WebSphere MQ , (2) olarak gösterilir. İstemci uygulaması, SOAP katmanına sahip `jms:` iletişim kuralı için bir işleyici olarak WebSphere MQ SOAP gönderen bileşenini kaydeder. SOAP katmanı, `jms:` ' a adreslenen SOAP iletilerini WebSphere MQ SOAP göndericisine aktarır. Gönderen, iletiyi istek kuyruğuna gerekli hizmet nitelikleri ile yerleştirmek için, iletide URI ' yi kullanır. Karşılık gelen WebSphere MQ SOAP dinleyicisi, istek kuyruğunda iletileri bekler ve istekleri işlemek ve yanıtları işlemek için SOAP katmanını çağırır.

SOAP gönderen ve dinleyici, olağan WebSphere MQ programlarıdır. Bunlar, Şekil 168 sayfa 914içinde olduğu gibi aynı kuyruk yöneticisine ya da farklı kuyruk yöneticilerine bağlı olabilir; bkz. Şekil 169 sayfa 915. İstemci bir istemci bağlantısıyla bağlanabilir.



Şekil 168. SOAP/WebSphere MQ tarafından kullanılan kuyruklar (tek kuyruk yöneticisi)



Şekil 169. SOAP/WebSphere MQ tarafından kullanılan kuyruklar (ayrı kuyruk yöneticileri)

SOAP ile JMS arasında bağ tanımlamaya ilişkinW3C aday önerisi.

W3C aday önerisi, SOAP üzerinde SOAP bağ tanımı (SOAP over Java Message Service 1.0) tanımlar. Ayrıca, bunun örnekleri için de yararlı olur: Java (tm) ileti hizmeti için URI şeması 1.0[11].

WebSphere Application Server v7gibi bazı uygulama çerçeveleri W3C aday önerisi için destek içerir. Send SOAP requests formatted with a URI compatible with the W3C candidate recommendation using the Axis2 client; see W3C SOAP over JMS URI for the WebSphere MQ Axis 2 client . The Axis2 client sends a SOAP request formatted with either a W3C or a WebSphere MQ transport for SOAP based on URI in the SOAP request.

W3C önerisine ilişkinAxis2 istemci desteği, 7.0.1.3 düzeltme paketinde sunulmuştur. Diğer istemciler için ve WebSphere MQ tarafından sağlanan SOAP dinleyicilerine ilişkin destek sağlanmaz.

İlgili kavramlar

SOAP for SOAP on .NET Framework 1, .NET 2 ve Axis 1.4için WebSphere iletimi somutlaması
Kendi WebSphere MQ SOAP göndericinizi ve dinleyicinizi yazmanız gerekebilir. .NET Framework 1, .NET Framework 2 ve Axis 1.4 üzerinde SOAP için WebSphere MQ iletimi somutlamasını kılavuz olarak kullanın.

SOAP ve Web hizmetleri güvenilir ileti sistemi içinWebSphere MQ iletimi
Web hizmetleri güvenilir ileti sistemi, güvenilir olmayan bir bağlantı üzerinden Web hizmeti isteklerini ve yanıtlarını güvenilir bir şekilde değiş tokuş etmek için kullanılan bir iletişim kuralıdır. Kısa ömürlü bağlantı kesintilerinin sorunları çözmek için en uygun çözümdür.

[11] En son taslağa ilişkin W3C belirtim başvurularında *JMS için URI Şeması*' na bakın.

SOAP for SOAP on .NET Framework 1, .NET 2 ve Axis 1.4için WebSphere iletimi somutlaması

Kendi WebSphere MQ SOAP göndericinizi ve dinleyicinizi yazmanız gerekebilir. .NET Framework 1, .NET Framework 2 ve Axis 1.4 üzerinde SOAP için WebSphere MQ iletimi somutlamasını kılavuz olarak kullanın.

1. Bir istemci programı, uygun Web hizmetleri çerçevesini, HTTP iletimi için olacağı şekilde kullanır. Ayrıca, `jms:` önekini kaydettirmelidir. Önek, `com.ibm.mq.soap.Register.extension()` Java yöntemi ya da `IBM.WMQSOAP.Register.Extension()` CLR yöntemi kullanılarak kaydedilmektedir.
2. Axis 1.4 ya da .NET Framework 1 ya da 2 framework, çağrıyı SOAP/HTTP için tam olarak SOAP isteği iletisine çağırmış olarak yayınlar.
3. A WebSphere MQ service is identified by a URI prefixed with `jms:`. When the framework identifies the `jms:` URI, it calls the WebSphere MQ transport sender code; `com.ibm.mq.soap.transport.jms.WMQSender` (for Axis 1.4) or `IBM.WMQSOAP.MQWebRequest` (for .NET1 and 2). Çerçeve, `http:` önekiyle bir URI saptarsa, HTTP üzerinden standart SOAP göndereni çağırır.
4. SOAP iletisi, istek kuyruğunu kullanarak WebSphere MQ SOAP göndericisi tarafından iletilir. The **`SimpleJavaListener`** (for Java) or **`amqwSOAPNETListener`** (for .NET) receives the request message.

   WebSphere MQ SOAP dinleyicileri, bağımsız süreçlerdir ve uyarlanabilir sayıda iş parçacıklı çok iş parçacıklı bir işlemdir.
5. WebSphere MQ SOAP dinleyicisi gelen SOAP isteğini okur ve uygun Web hizmeti altyapısına iletir.
6. Web hizmeti altyapısı, SOAP isteği iletisini ayrıştırır ve hizmeti çağırır. Yordam, HTTP iletisine gelen bir ileti için aynıdır.
7. Altyapı, yanıtı bir SOAP yanıt iletisine biçimlendirir ve bunu WebSphere MQ SOAP dinleyicisine döndürür.
8. Dinleyici iletiyi yanıt kuyruğuna yerleştirir ve ileti WebSphere MQ SOAP göndericisine aktarılır. Gönderen bunu istemci Web hizmeti altyapısına iletir.
9. İstemci altyapısı, yanıt SOAP iletisini ayrıştırır ve sonucu istemci uygulamasına geri el ile geri eder.

Her uygulama bağlamı, ayrı bir WebSphere MQ istek kuyruğu tarafından sunulur.

The application context is controlled in Axis 1.4 by ensuring that the WebSphere MQ SOAP listener and service execute in the appropriate directory. Axis 1.4 sets the correct CLASSPATH for the directory.

Application context is controlled in .NET by the WebSphere MQ SOAP listener executing the service in a context created by a call to `ApplicationHost.CreateApplicationHost`. Arama, hedef yürütme dizinini belirtir. Daha sonra, her hizmet konuşlandırıldığı dizinde çalışır.

`amqwdeployWMQService` , istek ve yanıt kuyruklarını oluşturur. Ayrıca, kuyrukların işlenmesi ve hizmetlerin Axis 1.4' e konuşlandırılması için gereken altyapıyı da oluşturur.

İlgili kavramlar

SOAP ve WebSphere MQ' nun bütünleştirilmesi

SOAP ve Web hizmetleri güvenilir ileti sistemi içinWebSphere MQ iletimi

Web hizmetleri güvenilir ileti sistemi, güvenilir olmayan bir bağlantı üzerinden Web hizmeti isteklerini ve yanıtlarını güvenilir bir şekilde değiş tokuş etmek için kullanılan bir iletişim kuralıdır. Kısa ömürlü bağlantı kesintilerinin sorunları çözmek için en uygun çözümdür.

SOAP ve Web hizmetleri güvenilir ileti sistemi içinWebSphere MQ iletimi

Web hizmetleri güvenilir ileti sistemi, güvenilir olmayan bir bağlantı üzerinden Web hizmeti isteklerini ve yanıtlarını güvenilir bir şekilde değiş tokuş etmek için kullanılan bir iletişim kuralıdır. Kısa ömürlü bağlantı kesintilerinin sorunları çözmek için en uygun çözümdür.

SOAP içinWebSphere MQ , SOAP iletilerini geçirmek için bir WebSphere MQ tarafından yönetilen ve güvenilir bir ağ kullanmaktan yararlanir. HTTP ve FTP gibi taşıma işlemi yönetilmeyen bir şekilde iletilmez. Yönetilmeyen ağlar, bağlantıların öngörülemeyen bağlantılar için idealdir. Bu bağlantılar, bağlantıların

yönetilmesine ilişkin zorlukların ve maliyetlerin, istek ve yanıtların kaybedilmemesinin avantajlarından daha ağır bir şekilde yararlanmalarıdır.

Yönetilmeyen ağlarda bağlantılar bozulduğunda dosyaların gevrmesi sorununun üstesinden gelmek için, yönetilen FTP gibi hizmetler, FTP ' nin üst kısmında bir yönetim katmanı oluşturur. Yönetim katmanı, dosyaların kullanıcılardan başarıyla aktarıldığını kontrol etme yükünü devralıyor ve gerekirse, eksik dosyaları yeniden iletiyor. Yönetilen FTP ' yi kullanmak için, bağlantının her iki ucunda da yönetim yazılımının kurulu olması gerekir.

Web hizmetleri güvenilir ileti sistemi (WSRM), güvenilir olmayan bağlantılar sorununu çözmek için farklı bir yaklaşım gerektirir. Bunun amacı, Web hizmeti isteklerini ve yanıtlarını güvenilir bir şekilde aktarmak ve her iki ucu da aynı yazılımı kullanmak zorunda kalmaksızın aktarmaktadır. Web hizmetleri güvenilir ileti alışverişi iletişim kuralını uygulayarak herhangi bir yazılım, diğer yazılımlarla güvenilir bir şekilde ileti alışverişi yapabilir.

Bir bağlantı başarısız olduğunda, üretilmiş bir URI ' yı anahtar olarak kullanarak, bir gönderen ve alıcı WSRM ileti aktarımının bağlamını korumalıdır. Gönderen ve alıcı yeni bir bağlantı kurmayı deniyor. Yeni bir bağlantı başarıyla kurulduysa, aktarma işlemi tamamlanır. WSRM belirtimi, bağlamın nasıl korunmuş olduğunu ya da yeni bir bağlantı denendiğinde belirtilmez.

Sadece kısa süreli kesintilerin ilgi çekeceğine karar verebilirsiniz. Daha uzun kesintiler için, bir süre sonra yeniden başlatılamayabilecek aktarımları atmak üzere hazırlanabilirsiniz. Benzer şekilde, istemci ya da hizmet başarısız olursa, aktarma işlemini iptal etmeye hazır olabilirsiniz. Aktarımları sağlamak için kullanıcıyı sorumlu bırakmak, müşteri ve hizmetin koordinasyonunu yönetmeye ilişkin daha az talep görmektedir.

Ağ kesintileri uzun ömürlü olduğunda, 30 dakikadan fazla ya da istemci ya da sunucu arızalanırsa, bazı bağlantıların yeniden oluşturulamaması olasılığı artar. Yönetilen bir şekilde, WSRM ' nin ileti aktarımında otomatik olarak otomatik olarak geri yüklenmesine güvenilmez. Başarısız olan WSRM bağlantılarını yönetmeyi göz önünde bulundurmanız gerekir. Bu bağlantılar, müşterilerin ve hizmetlerin ağını yönetmek için yazılım geliştirmeyi anlamına gelir.

Kısa kesintileri aşmak için WSRM ' nin kullanılması, mobil bir ağdaki kayıp iletilerle ilgili olarak önemli ölçüde azaltabilir. İleti teslimi temin etmek zorunda değilseniz, ileti kaybını azaltmanın yararları, bir WSRM somutlamasının geliştirilmesinin ek maliyetini haklı çıkarabilirler.

SOAP over JMS, istemciye, sunucuya ve ağa ilişkin uzun süreli kesintiler ile güvenli ileti teslimi ve anlaşmaları sağlar. SOAP için HTTP ' ye göre daha güvenilir bir hizmet kalitesi arıyorsanız, hangi çözümü seçiniz: SOAP ya da WSRM için WebSphere MQ iletimi? Cevap birçok faktöre bağlı. Göz önünde bulundurulması gereken bazı etkenler şunlardır:

1. Güvenilirlik, bağlantı hatasından kaynaklanıp kaynaklanmadığı.
2. Bağlantı başarısızlıklarının ne kadar uzun olduğu.
3. Bağlantının hem istemci hem de sunucu tarafını yönetebilrseniz.
4. Kullanıcı ya da bir yönetici, iletilerin tesliminden en nihayetinde sorumlu olur.

İlgili kavramlar

SOAP ve WebSphere MQ' nun bütünleştirilmesi

SOAP for SOAP on .NET Framework 1, .NET 2 ve Axis 1.4için WebSphere iletimi somutlaması

Kendi WebSphere MQ SOAP göndericinizi ve dinleyicinizi yazmanız gerekebilir. .NET Framework 1, .NET Framework 2 ve Axis 1.4 üzerinde SOAP için WebSphere MQ iletimi somutlamasını kılavuz olarak kullanın.

WebSphere MQ Web hizmetlerinin kurulması ve doğrulanması

SOAP için WebSphere MQ iletimi olanağını kurmak ve doğrulamak için bu konularda yönergeleri kullanın.

SOAP için WebSphere MQ Web iletimi kurulması

SOAP için WebSphere MQ Web iletimi olanağını kurmak için bu yönergeleri kullanın. Kuruluş, SOAP iletimi olarak WebSphere MQ ' yı kullanarak Web hizmeti istemcilerini ya da hizmetlerini çalıştırmak için araçlar yaratır. Araçlar, .NET Framework 1, .NET 2, Axis 1.4ya da Axis2 SOAP ortamlarında kullanılır.

Başlamadan önce

Önkoşul olan ürünleri IBM WebSphere MQiçin Sistem Gereksinimleriadresinden denetleyin. Kuruluş işlemi, önkoşul olan yazılımların varlığını ve kullanılabilirliğini denetmiyor. Önkoşulların kurulu olduğunu doğrulamanız gerekir.

WebSphere MQ , Axis 1.4 yürütme zamanından oluşan bir kopya sağlar. Kurmuş olabileceğiniz herhangi bir yerine WebSphere MQ ile bu sürümü kullanın. IBM , Apache Axis için teknik destek sağlamaz. Teknik sorunlarınız varsa, Apache Software Foundation ile iletişim kurun.

Web hizmetlerini .NET Framework 3 SOAP ortamında çalıştırmak için, WebSphere MQ , Windows Communication Foundation 'ı kullanır. WebSphere MQ özel kanalı için Windows Communication Foundation, SOAP iletileri için bir iletim olarak WebSphere MQ olanağını kullanarak Web hizmeti istemcilerini ve hizmetlerini çalıştırır.

Bu görev hakkında

You can install the WebSphere MQ Web transport for SOAP as either a WebSphere MQ MQI client or Server application. WebSphere MQ ürününü bir istemci olarak ya da bilgisayarınızda bir sunucu olarak önceden kurdıysanız, listelenen bileşenleri kurmadığınızı denetleyin.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

Aşağıdaki kuruluş adımlarını gerçekleştirebilirsiniz.

Yordam

1. Kuruluş için "Java ve. Net Messaging and Web services" bileşenini seçin.
2. Solaris ve HP-UXüzerinde kuruluş için "Java Runtime Environment" bileşenini seçin.
3. Kuruluş için Development Toolkit 'i seçin.
4. Platformunuza ilişkin Quick Beginning içinde açıklandığı gibi WebSphere MQ ürününü kurun ve doğrulayın.
5. Apache Axis 1.4 yürütme zamanını, `axis.jar` , WebSphere MQ kuruluş ortamının `prereqs/axis` dizininden kopyalayın. Bunu, Çizelge 138 sayfa 919, Çizelge 139 sayfa 919 ya da Çizelge 140 sayfa 919içinde açıklanan kuruluş dizinine kopyalayın.

   Pencereler

   ```
   Copy D:\PreReqs\axis\axis.jar MQ_INSTALLATION_PATH\java\lib\soap
   ```

   AIX

   ```
   cp -f PreReqs/axis/axis.jar MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   chown mqm:mqm MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   chmod 444 MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   ```

   HP-UX, Solaris ve Linux (tüm platformlar) kuruluş dizinleri

   ```
   cp -f PreReqs/axis/axis.jar MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   chown mqm:mqm MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   chmod 444 MQ_INSTALLATION_PATH/java/lib/soap/axis.jar
   ```

6. Pencereler 2003 'te komut dosyası eşlemlerini, kullandığınız Ortak Dil Çalıştırma zamanı sürümüne işaret edecek şekilde güncellemek için **Aspnet_regiis.exe** komutunu çalıştırın.

   %SystemRoot%Microsoft.NET/Framework/*version-number* içindeki **Aspnet_regiis.exe** yardımcı programını arayın.
7. Set the environment variable, WMQSOAP_HOME , to point to the WebSphere MQ installation directory.

Sonuçlar

Çizelge 138. Windows kuruluş dizinleri	
Konum	**İçindekiler**
MQ_INSTALLATION_PATH\programs\bin	İkili, komut, DLL ve yürütülür dosyalar
MQ_INSTALLATION_PATH\programs\java\lib	.jar files
MQ_INSTALLATION_PATH\programs\java\lib\soap	SOAP .jar files
MQ_INSTALLATION_PATH\programs\soap\samples	Örnekler ve IVT

Çizelge 139. AIX kuruluş dizinleri	
Konum	**İçindekiler**
MQ_INSTALLATION_PATH/bin	Kabuk komut dosyaları
MQ_INSTALLATION_PATH/java/lib	.jar files
MQ_INSTALLATION_PATH/java/lib/soap	axis.jar ve diğer JAX-RPC .jar dosyaları
MQ_INSTALLATION_PATH/samp/soap	Örnekler ve IVT

Çizelge 140. HP-UX, Solaris ve Linux (tüm platformlar) kuruluş dizinleri	
Konum	**İçindekiler**
MQ_INSTALLATION_PATH/bin	Kabuk komut dosyaları
MQ_INSTALLATION_PATH/java/lib	.jar files
MQ_INSTALLATION_PATH/java/lib/soap	axis.jar ve diğer JAX-RPC .jar dosyaları
MQ_INSTALLATION_PATH/samp/soap	Örnekler ve IVT

Sonraki adım

1. Yalnızca .NET için, SOAP dosyaları için WebSphere MQ iletimi için Genel Derleme Önbelleğini kaydetmeniz gerekir. WebSphere MQ' u kurduğunuzda .NET zaten kuruluysa, kayıt otomatik olarak kuruluş sırasında gerçekleştirilir. .NET 'i WebSphere MQ' dan sonra kurduğunuzda, IVT ilk çalıştırıldığında kayıt otomatik olarak gerçekleştirilir.

   .NET yapıbirimlerine kayıt gerçekleştirmek için **amqiregisterdotnet.cmd** komutunu çalıştırabilirsiniz. Ayrıca, herhangi bir aşamada yeniden kayıt işlemi yapmak için **amqiregisterdotnet.cmd** komutunu çalıştırabilirsiniz. Bu kayıt bir kez yapıldığında, sistem yeniden başlatıldıktan sonra sistem yeniden başlatıldıktan sonra da olağan bir şekilde yeniden kayıt işlemi gerekmez.
2. Run the Installation Verification Test, as described in “SOAP için IBM WebSphere MQ iletimi doğrulanıyor” sayfa 919.
3. Axis2 istemcisi geliştirmeyi düşünüyorsanız, Axis2 1.4.1 dosyasını Apache' den yüklemelisiniz; bkz. “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942.

SOAP için IBM WebSphere MQ iletimi doğrulanıyor

Verify the IBM WebSphere MQ transport for SOAP using the **runivt** command. Komut, bir dizi gösterim uygulamasını çalıştırır ve kuruluştan sonra ortamın doğru bir şekilde ayarlanmasını sağlar.

Başlamadan önce

runivt komutunu çalıştırmadan önce, aşağıdaki yürütme ortamlarının olduğundan emin olun:

- Yalnızca Axis 'te çalıştırmak için: sisteminizde kullanılabilir bir Java SDK (SOE içinde) olmalıdır. Ayrıca, sistem **PATH** ortam değişkeninde `java.exe` ve `javac.exe` komutlarının yerini de eklemelisiniz.
- Sınamayı yalnızca .NET üzerinde çalıştırmak için (yalnızca Windowsüzerinde desteklenir): sisteminizde hem Java SDK, hem de .NET derleyiciler ve araçlar olmalıdır. Bunu yapmak için, bir Visual Studio komut istemine ya da Microsoft Windows SDK komut istemine erişin, ardından `java.exe` ve `javac.exe` dosyalarının konumunu **PATH** ortam değişkenine ekleyin.
- Kullanılabilir tüm sınamaları çalıştırmak için: Windows altyapılarında, ortam .NET test çalıştırmasında açıklandığı şekilde yapılandırılmalıdır. UNIX and Linux platformlarında, ortamın yalnızca Axis test çalıştırmasında açıklandığı şekilde yapılandırılması gerekir.

Bu görev hakkında

Doğrulama sınamasını hem .NET hem de Axis 'te çalıştırmak yerine, sınamayı yalnızca Axis 'de ya da .NET üzerinde çalıştırmak isteyebilirsiniz.

Sınamalarla ilgili sorun yaşarsanız ve yeniden başlamak istiyorsanız:

1. Stop the queue manager `WMQSOAP.DEMO.QM` using the `hemen` option.
2. Farklı bir pencerede başlatılmış olan dinleyiciyi durdurun.
3. Kuyruk yöneticisini silin.
4. Yarattığınız geçici örnekler dizinini silin ve yeniden başlatın.

UNIX and Linux platformlarında, bir X Windows sistem oturumu kullanarak komutu çalıştırmanız gerekir.

runivt komutu, `soap/samples` dizininin içeriğini değiştirir. Kuruluş görüntüsünü değiştirmeden alıkoymak için, örnek dizinini geçici bir konuma kopyalayın ve geçici konumdan doğrulama sınamasını çalıştırın.

Kuruluş doğrulamayı istediğiniz sayıda çalıştırabilirsiniz.

Carry out the following steps to verify the installation of IBM WebSphere MQ transport for SOAP on .NET Framework 1, .NET Framework 2, and Axis 1.4:

Yordam

1. `./tools/soap/samples` dizin ağacını geçici bir konuma kopyalayın.
2. Geçici dizini geçerli dizin olarak kullanarak bir komut penceresi başlatın.
3. Kuruluş sınamasını başlatmak için **runivt** komutunu kullanın. `runivt` komut dosyası, bir test sınıfını, örnek istemciyi ve hizmetleri konuşlandırmadan ve çalıştırmadan önce derler. Sınama sınıfı, örnek istemci ve çalıştırılacak hizmetler için, SOAP için WebSphere(r) MQ Web iletimi kuruluyor içinde belirtilen kuruluş adımlarını tamamlayın ve `runivt` komutunu çalıştırmak için kullanılan komut isteminin gerekli yürütme ortamı kümesine sahip olduğundan emin olun. **runivt** komutunu çalıştırmak için aşağıdaki yöntemlerden birini kullanın:
   - Yalnızca Axis 'te bir test çalıştırın: `runivt Axis`.
   - Sınamayı yalnızca .NET üzerinde çalıştırın (yalnızca Windowsüzerinde desteklenir): `runivt DotNet`.
   - Tüm kullanılabilir sınamaları çalıştırın:`runivt`.

   `runivt` komut sözdizimi ve değiştirgeleriyle ilgili daha fazla bilgi için bakınız: **runivt**: IBM WebSphere MQ transport for SOAP installation verification test. Çalıştırabileceğiniz sınamalar, Windows üzerinde `ivttests.txt` ve UNIX and Linux platformlarında `ivttests_unix.txt` dosyasında listelenir.

İlgili başvurular

runivt: SOAP kuruluşu doğrulama sınaması için WebSphere MQ iletimi

SOAP için WebSphere MQ iletimi için Web hizmetleri geliştirilmesi

SOAP için WebSphere MQ iletimi ile kullanmak üzere hizmetler geliştirmek için normal Web hizmeti geliştirme ortamınızı kullanın.

Başlamadan önce

1. SOAP için WebSphere MQ iletimi ile birlikte verilen komut satırı araçlarını kullanmayı planlıyorsanız:
   a. Hizmet için bir konuşlandırma dizini oluşturun.
   b. Dizinde bir komut penceresi başlatın.
   c. .NET için, csc.exe ve wsdl.exe yolunda olmalı ve .NET Framework 'un aynı sürümlerinden olmalıdır.
   d. Java için,
      i) Sınıf yolunu (classpath) ayarlamak için **amqwsetcp** komutunu çalıştırın.
      ii) Aynı sürüm düzeyinde bir IBM JRE ve JDK geçerli yolda olmalıdır. Sürüm düzeyi en az 5.0olmalıdır.
      iii) Sınıf yolunu, geliştirmekte olduğunuz hizmet için de dahil olmak üzere, ek `.jar` kitaplıklarının ve .java paketlerini içeren dizinlerin yer almasını içerecek şekilde uyarlayın. Yürürlükteki dizini (classpath) "`.`" dizinine koyun.
      iv) Komut penceresinin yürürlükteki diziniyle göreli olarak, geliştirmekte olduğunuz hizmetin paket adına karşılık gelen bir dizin yaratın.
2. Diğer bir seçenek olarak, Web hizmetleri geliştirmesini destekleyen çalışma ortamı araçlarını kullanın. Örnek geliştirme görevleri, Microsoft Visual Studio 2008, Eclipse IDE for Java EE Developers ve WebSphere Application Server Community Edition' i kullanır.

Bu görev hakkında

Var olan Web hizmetlerinin, SOAP için WebSphere iletimiyle çalışmak için herhangi bir değişiklik yapılmaması gerekir. SOAP için WebSphere MQ iletimi ile sağlanan araçlar bir Web hizmetini konuşlandırr ve WebSphere MQ SOAP dinleyicisi kullanarak çalıştırır. Araçlar, SOAP istemcileri için WebSphere MQ iletimi geliştirmek için WSDL, .NET istemcisi sınırlı kod öbekleri ve `.java` yetkili sunucusu sınıfları da oluşturur.

Bir hizmet oluşturmak için bu adımları izleyin ve bunu, devreye alma ve istemci oluşturma için hazırlayın. Eclipse ya da Microsoft Visual Studio 2008 kullanarak bir hizmet yaratmak için ilgili görevlerdeki adımları izleyin.

Yordam

1. Olağan geliştirme ortamınızı kullanarak hizmeti geliştirin.
2. HTTP Web hizmetleri istemcisini kullanarak hizmeti test edin
3. Konuşlandırma dizinini hazırlamak için aşağıdaki adımları izleyin:
   - Java için
     a. Hizmet arabirimini tanımlayan `.java` dosyasını konuşlandırma dizinine kopyalayın.
     b. Hizmet için herhangi bir `.class` dosyasını, paket adına karşılık gelen dizine kopyalayın.
     c. Check that the classpath can locate all the classes that are required: compile the service `.java` file using **javac**.
   - .NET için
     a. Hizmeti tanımlayan `.asmx` dosyasını konuşlandırma dizinine kopyalayın ve
     b. Kod arkasındaki modeli kullanıyorsanız, herhangi bir `.dll` dosyasını bir *deployment directory*`\bin` dizinine kopyalayın.

Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse

Hizmet sağlayıcınız olarak WebSphere MQ ' yı kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti geliştirin. 1.4eksenine yerleştirmeye yönelik bir hizmet oluşturmak için normal Web hizmeti geliştirme ortamınızı kullanın.

Başlamadan önce

Take into account the requirements for deploying a Web server to the WebSphere MQ SOAP listener for Axis 1.4.

- The WebSphere MQ SOAP listener for Axis 1.4 requires an IBM JRE at version 5.0 or greater. Geliştirme için kullanılan JRE ve JDK, aynı sürüm düzeyinde olmalıdır.
- The WebSphere MQ SOAP listener for Axis 1.4 requires the `axis.jar` installed with WebSphere MQ. Geliştirme ortamınızdaki oluşturma yolunu, geliştirme ortamıyla kurulan `axis.jar` dosyaları yerine, WebSphere MQile kurulan `axis.jar` dosyasına gönderme yapacak şekilde değiştirin.
- WebSphere MQ V7.0.1de dahil olmak üzere, devreye alınan hizmet için oluşturulan WSDL RPC/kodlanır. V7.1'den, RPC/hazır stil WSDL' yi de isteyebilirsiniz. Oluşturulan WSDL yalnızca konuşlandırma için kullanılır. Hizmeti WS-I uyumlu WSDL ' yi kullanarak tanımlayabilirsiniz.

Bu görev hakkında

Olağan Web hizmetleri geliştirme ortamınızı kullanarak hizmeti yaratın.

Bu görevde, Galileo olarak bilinen Web Geliştiricileri için serbestçe kullanılabilir açık kaynak Eclipse Java EE IDE ' yi kullanırız. Uygulama sunucusu için, Geronimo 'ya dayalı olarak WebSphere Application Server Community Edition v2.1 (Community Edition) olanağını kullanırız. IDE ve sunucunun nasıl edinileceği, kurulacağı ve yapılandırılacağı hakkında bilgi için ilgili görevlere bakın.

Daha önce IDE ' de sağlanan Web Hizmetleri Gezgini 'ni kullanarak bir iletim olarak HTTP kullanarak hizmeti test edin. Alternatif olarak, bir HTTP istemcisi yetkili sunucusu oluşturun ve kendi istemci kodunuzu kullanarak hizmeti test edin.

Bir alt Web hizmeti geliştirmek için bu adımları izleyebilirsiniz. Örnek olarak `StockQuoteAxis.java` örnek programını kullanın.

Yordam

1. Yeni bir çalışma alanıyla Java EE geliştiricileri için Eclipse IDE ' yi başlatın.
2. Çalışma alanını Java50kullanacak şekilde yapılandırın.
   WebSphere Application Server Community Edition 2.1.4 , Java60ile çalışmaz.
   a) **Pencere** > **Tercihler** > **Java** > **Kurulu JRE ' ler** > **Ekle ...** > **Standart VM** > **İleri** > **Dizin ...**
   b) Browse to the install directory of **Java50** > **Tamam** > **Son**
   c) **Java50** JRE > OK (Tamam) seçeneğini işaretleyin
3. Community Edition çalıştırma zamanı ortamını ekleyin ve Community Edition' ı başlatın.
   a) **Pencere** > **Tercihler** > **Sunucu** > **Çalıştırma Zamanı Ortamları** > **Ekle ...**
   b) **New Server Runtime Environment** (Yeni Sunucu Yürütme Ortamı) listesinden **IBM WASCE v2.1** öğesini seçin > **Yeni yerel sunucu yarat** > **İleri**seçeneğini belirleyin.
      IBM WASCE 2.1 listede yoksa, diğer iki görevi tamamlamanız gerekir:
      i) WebSphere Application Server Community Edition' ı kurun.
      ii) Community Editioniçin Eclipse güncellemesini kurun.
      Ayrıntıları WebSphere Application Server Community Editionadresinde bulabilirsiniz.
   c) Application Server Installation Directory > **OK** > **Finish** > **OK**(Tamam) > Uygulama Sunucusu Kuruluş Dizinine göz atın.
   d) Sunucular görünümünde **IBM WASCE v2.1 sunucusu** seçeneğini sağ tıklatın > **Başlat**

İpucu: You can manage WASCE in Eclipse: Right-click **IBM WASCE v2.1 sunucusu** > **WASCE Konsolunu Başlat** . Varsayılan **Username** ve **Password** , `system` ve `manager` ' dir.

4. Web hizmetleri için sunucuyu ve yürütme ortamını ayarlayın.
   a) **Pencere** > **Tercihler** > **Web hizmetleri** > **Sunucu ve Çalıştırma Zamanı**
   b) Sunucu olarak **IBM WASCE v2.1 Server** ' ı seçin.
   c) Web hizmeti yürütme ortamı olarak **Apache Axis** ' i bırakın.
5. Devingen bir Web projesi yaratmanızı sağlar.
   a) **Dosya** > **Yeni** > **Dinamik Web Projesi**.

      `StockQuoteAxis`projesinin adını yazın.
   b) **Bir EAR için proje ekle** > **Yeni ...**seçeneğini işaretleyin.
   c) **EAR Application Project** (EAR Uygulaması Projesi) sayfasında **Project name** `StockQuoteAxisEAR` > **Finish**(Son) sayfasını yazın

      Reply **Tamam** in response to the dialog box suggesting you switch to the Java EE perspective, or stay Java perspective to follow these instructions exactly.
   d) Hedef yürütme ortamı olarak**IBM WASCE 2.1 sunucusu** seçildi. Bunu kabul edin ve diğer varsayılan değerleri > **Bitir**' i tıklatın.

      Reply **Tamam** in response to the dialog box suggesting you switch to the Java EE perspective, or stay Java perspective to follow these instructions exactly.
6. `StockQuoteAxis.java` örnek programını içe aktarın
   a) Open the **StockQuoteEkseni** Web project > Right click the **src** folder > **İçe Aktar ...**
   b) **Genel** > **Dosya Sistemi** > **İleri**seçeneklerini belirleyin.
   c) *MQ_INSTALLATION_PATH*`\tools\soap\samples\java\server` > şuna göz atın: **StockQuoteAxis.java** > **Son**

      MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder. İçerdiği dosyaları görmek için sunucu dizinini vurgulamanız gerekir.
7. `StockQuoteAxis.java` öğesini doğru paketiyle taşıyarak derleme hatasını düzeltin.
   a) `StockQuoteAxis.java` ' u açın ve sorunu sağ tıklatın > **Hızlı Düzeltme**
   b) Double click **'StockQuoteAxis.java' paketini 'soap.server' paketine taşı** > **Kaydet**.
8. `StockQuoteAxis.java` ' dan bir Web hizmeti oluşturun.
   a) `StockQuoteAxis.java` > **Web Services** > **Create Web service** > **Next**(Web Hizmetleri Oluştur > İleri) seçeneklerini sağ tıklatın.

      Hizmet için varsayılan yapılandırmayı kabul edin:

      Web hizmeti tipi
      Java beanWeb Service 'i alt üst

      Hizmet somutlaması
      soap.server.StockQuoteAxis

      Sunucu
      `IBM WASCE v2.1 server`

      Web hizmeti yürütme ortamı
      Apache Ekseni

      Hizmet projesi
      StockQuoteEkseni

      Hizmet EAR projesi
      StockQuoteAxisEAR

      Yapılandırma
      İstemci oluşturma yok
9. Erişilecek yöntemleri ve Web hizmeti stiline (**İleri**) seçin.

   İstenirse, sunucuyu başlatın.

a) Tüm yöntemleri seçili olarak bırakın.

b) Belge/hazır bilgi (sarılmış) stili seçin.

10. **Son**

    Hizmet konuşlandırıldığında, `StockQuoteAxis` Web projesindeki `WebContent\wsdl` klasörüne bakın ve oluşturulan `StockQuoteAxis.wsdl` dosyasını bulun.

11. Web Services Explorer ile HTTP kullanarak hizmeti test edin.

    a) `StockQuoteAxis.wsdl` > **Test with Web Services Explorer**(Web Hizmetleri Gezgini ile Test) seçeneğini sağ tıklatın.

    b) **Web Hizmetleri Gezgini** penceresindeki **StockQuoteAxisSoapBinding** işlemlerindeki **getQuote** işlemini tıklatın.

    c) **Simge** giriş alanına `ibm` yazın > **Git**

12. SOAP için WebSphere MQ iletimi kullanarak hizmeti test edin.

    Hizmeti konuşlandırmak için, `com.ibm.mq.soap.jar`' ta bulunan **`SimpleJavaListener`**dosyasını kullanın. Oluşturma yoluna WebSphere MQ Java ve SOAP kitaplıklarını eklememeniz gerekir.

    a) **StockQuoteAxis** Web project > **Build Path** > **Configure Build Path ...**öğelerini sağ tıklatın.

    b) **Kitaplıklar** sekmesini > **Dış Jar Ekle ...**seçeneğini tıklatın. *`MQ_INSTALLATION_PATH`*`\java\lib` ' a göz atın. ve tüm `.jar` dosyalarını > **Aç** > **Dış Jar Ekle ...**seçeneklerini belirleyin. *`WMQ Install directory`*`\java\lib\soap` ' a göz atın ve tüm `.jar` dosyalarını > **Aç** > **Tamam**seçeneklerini belirleyin.

    `MQ_INSTALLATION_PATH` , WebSphere MQ ' un kurulu olduğu üst düzey dizini temsil eder.

    c) Proje Gezgini 'nde, `StockQuoteAxis\Java Resources\Libraries\com.ibm.mq.soap.jar\com.ibm.mq.soap.transport.jms\SimpleJavaListener.class\ SimpleJavaListener` > **Olarak çalıştır ...**> **Yapılandırmaları Çalıştır ...**öğelerini seçin.

    İpucu:

    > `SimpleJavaListener` için herhangi bir yapılandırma yoksa, **Çalıştırma yapılanışları** sihirbazının **Yapılandırma oluştur, yönet ve çalıştırma yapılanışları** sayfasındaki **Yeni yapılandırma** simgesini tıklatın.
    > **`SimpleJavaListener`** ' in bunu durdurabilmek için hiçbir komutu yoktur. **`SimpleJavaListener`**programını izlemek ya da durdurmak için, Eclipseiçinde **Hata ayıklama perspektifi** ' i açın.

    d) **(x) = Bağımsız Değişkenler** sekmesini açın. **Program bağımsız değişkenleri** giriş alanında, parametreleri **`SimpleJavaListener`**olarak yazın.

    Bu örnek için şunu yazın:

    ```
    -u "jms:/queue?destination=REQUESTAXIS@QM1&connectionFactory=()
    &initialContextFactory=com.ibm.mq.jms.Nojndi
    &targetService=StockQuoteAxis
    &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE" -n 10
    ```

    Not: Hedef hizmet, hizmet konuşlandırma tanımlayıcısında (`StockQuoteAxis\WebContent\WEB-INF\server-config.wsdd`) oluşturulan hedef hizmet adıyla eşleştirmek için `StockQuoteAxis` . amqwdeployWMQService , `soap.server.StockQuoteAxis`adlı bir hedef hizmet yaratır. Bu örnekte, HTTP sunucusu olarak aynı `StockQuoteAxis.class` ve `service-config.wsdd` ' yi kullanıyorsunuz.

    e) Aynı etikette, `server-config.wsdd` dosyasına gönderme yapmak için **Çalışma dizini** ' yi yapılandırın: `${workspace_loc:StockQuoteAxis/WebContent/WEB-INF}`

    a) **Çalıştır**

    Hatalar konsola yazılır. Konsol boş kalırsa, **`SimpleJavaListener`** ' un durumu iyi olur.

b) To test the deployment, run a StockQuoteAxis client, developed in the task, “Developing a JAX-RPC client for WebSphere transport for SOAP using Eclipse” sayfa 935.

Örnek: StockQuoteAxis örnek programı

Örnek Java Web hizmeti (`StockQuoteAxis.java`), *WMQ install directory*`\tools\soap\samples\java\server` içinde kurulur. `StockQuoteAxis.java`, Şekil 170 sayfa 925, dört yönteme sahiptir:

1. `float getQuote(String symbol)`
2. `void getQuoteOneWay(String symbol).`
3. `int asyncQuote(int delay)`
4. `float getQuoteTran(String symbol)`

```
package soap.server;
import java.lang.Thread;
import java.io.FileWriter;
public class StockQuoteAxis {
    public float getQuote(String symbol) throws Exception {
        return ((float) 55.25);
    }
    public void getQuoteOneWay(String symbol) throws Exception {
        try {
            // Write the results for this service to a file
            FileWriter f = new FileWriter("getQuoteOneWay.txt", true);
            f.write("One way service result via proxy is: 44.44\n");
            f.close();
        } catch (Exception ee) {
            System.out.println("Error writing result file in getQuoteOneWay");
            ee.printStackTrace();
        }
    }
    public int asyncQuote(int delay) {
        try {
            Thread.sleep(delay);
        } catch (Exception e) {
            System.out.println("Exception in asyncQuote during sleep");
        }
        return delay;
    }

    public float getQuoteTran(String symbol) throws Exception {
        if (symbol.equalsIgnoreCase("ROLLBACK")) {
            System.out.println("Rollback was requested,
                exiting from service by calling System.exit().");
            System.exit(0);
        }
        return ((float) 55.25);
    }
}
```

Şekil 170. StockQuoteEkseni

Sonraki adım

Deploy the service using WebSphere MQ Transport for SOAP, instead of HTTP using the command **amqwdeployWMQService**.

The command has an option, `axisDeploy`, which deploys the service by creating an Apache Axis 1.4 deployment descriptor. WebSphere MQ SOAP dinleyicisi hizmeti çalıştırır. SOAP dinleyicisine SimpleJavaDinleyicisi adı verilir ve SOAP için WebSphere MQ iletimi ile birlikte sağlanır.

İlgili görevler

Developing a .NET 1 or 2 service for WebSphere MQ transport for SOAP using Microsoft Visual Studio 2008

Develop the SampleStockQuote Web service for .NET 1 or .NET 2 using Microsoft Visual Studio 2008

JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a JEE application server. This task is step 2 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol.

Developing a .NET 1 or 2 service for WebSphere MQ transport for SOAP using Microsoft Visual Studio 2008

Develop the SampleStockQuote Web service for .NET 1 or .NET 2 using Microsoft Visual Studio 2008

Bu görev hakkında

Visual Studio 2008 kullanarak bir kod arkalı somutlama ile StockQuote hizmetini yaratın.

Yordam

1. Hizmet için bir şablon oluşturun ve HTTP ' de çalıştırdığını doğrulayın.
   a) Visual Studio 2008 > **Dosya** > **Yeni** > **Proje ...**öğelerini başlatın. **C#** Proje Tipi, **, NET Framework 2**ve **ASP.NET Web Hizmeti Uygulaması**. **Name:** (Ad) ve **Solution Name (Çözüm Adı):** `StockQuoteDotNet` > **OK**(Tamam) yazın
   b) **Çözüm Gezgini** > **Yeniden Adlandır** > `StockQuote.asmx`içinde **Service1.asmx** öğesini farenin sağ düğmesiyle tıklatın.
   c) `public class Service1` kod parçasını `public class StockQuote`olarak değiştirin.
   d) **Çözüm Gezgini** > **Birlikte Aç ...** > **XML Düzenleyicisi**' nde **StockQuote.asmx** öğesini farenin sağ düğmesiyle tıklatın. `Class="StockQuoteDotNet.Service1"` seçeneğini `Class="StockQuoteDotNet.StockQuote"`olarak değiştirin
   e) `[WebService(Namespace = "http://tempuri.org/")]` kod parçasını `[WebService(Namespace = "http://stock.samples/")]`olarak değiştirin.
   f) Remove the line of code `[ToolboxItem(false)]`.
   g) Şu ana kadar her şeyin doğru olup olmadığını denetleyin: **Hata Ayıklama** > **Hata Ayıklamayı Başlat (F5)**. Explorer 'da çıktıyı doğrulayın.
2. Yöntemleri örnek `SQDNNonInline.asmx.cs`'den ekleyin ve HTTP' de hizmeti test edin.
   a) Open *MQ_INSTALLATION_PATH*`\tools\soap\samples\dotnet\SQDNNonInline.asmx.cs` and replace the `HelloWorld` method with the four `Quote` methods; see Şekil 171 sayfa 928. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizini temsil eder.
   b) **Oluşturma** > **Yeniden Oluştur** the solution > Right click one of the **İş Parçacığı** in error > **Çözümle** > Using **System.Threading**.
   c) Hata ayıklamayı başlatmak için F5 tuşuna basın.

      Hizmet WS-I Basic Profile v1.1' e uyumlu değil. WebMethod ek açıklamasını `[SoapRpcMethod]` 'dan `[SoapDocumentMethod]`' a değiştirmek ya da `[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]`ek açıklamasını kaldırmak seçeneğiniz vardır.
   d) HTTP ' yi kullanarak uygulamayı doğrulamak için F5 tuşuna basın.
3. WSDL ' yi, istemcileri oluşturun ve SOAP için WebSphere MQ iletimi kullanarak hizmeti çalıştırın.
   a) Proje dizini ağacındaki bir komut penceresi açın; burada `StockQuote.asmx` saklanır.
   b) (İsteğe bağlı) Yapay nesneler oluşturmak için amqwdeployWMQService öğesini kullanın. Kuyruk yöneticisi başlatılmalı:

      ```
      amqswdeployWMQService -f StockQuote.asmx
      -u "jms:/queue?initialContextFactory=com.ibm.mq.jms.Nojndi
      &connectionFactory=()
      &destination=REQUESTDOTNET@QM1
      &targetService=StockQuote.asmx"
      StockQuote.asmx StockQuote.wsdl
      ```

      Tüm yapay nesneler, `./generated` dizin ağacında oluşturulur.

c) (İsteğe bağlı) SOAP için WebSphere MQ iletimi kullanan hizmeti çağırmak için yalnızca WSDL ' yi oluşturun.

```
amqswsdl -u "jms:/queue?initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=()
&destination=REQUESTDOTNET@QM1
&targetService=StockQuote.asmx"
StockQuote.asmx StockQuote.wsdl
```

a) .NET dinleyicisini çalıştırın. `.\generated\server\startWMQNListener.cmd` komutunu kullanın ya da komutu yazın:

```
amqSOAPNETListener -u "jms:/queue?initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=()
&destination=REQUESTDOTNET@QM1
&targetService=StockQuote.asmx"
```

4. WSDL ' den oluşturulan bir istemciyi kullanarak ya da **amqwdeployWMQService**tarafından oluşturulan istemcileri kullanarak hizmeti test edin.

Örnek kod

Örnek .NET Web hizmeti (`StockQuoteDotNet`), *MQ_INSTALLATION_PATH*`\tools\soap\samples\dotnet`' ta kurulur. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir. Yayınlanan örneklerin Web hizmeti bağ tanımı, görevde kullanılan bağlayıcından biraz farklıdır. Görev, Visual Studio 2008 'de kullanılan varsayılanları kullanır.

.NET Framework 1 ve .NET Framework 2 Web hizmetlerine ilişkin iki örnek vardır. `StockQuoteDotNet.asmx`, yerleşik bir hizmettir. `SQDNNoninline.asmx`, `SQDNNoninline.asmx.cs`tarafından uygulanan bir kod arkasındaki Web hizmetidir.

`StockQuoteDotNet` ' in dört yöntemi vardır:

1. `float getQuote(String symbol)`
2. `void getQuoteOneWay(String symbol).`
3. `int asyncQuote(int delay)`
4. `float getQuoteDOC(String symbol)`

```
<%@ WebService Language="C#" Class="StockQuoteDotNet" %>
using System;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Web.Services.Description;
using System.Threading;
[WebService (Namespace="http://stock.samples")]
public class StockQuoteDotNet {
     [WebMethod] [ SoapRpcMethod(OneWay=true) ]
     public void getQuoteOneWay(String symbol) {
          if (symbol.ToUpper().Equals("DELAY")) Thread.Sleep(5000);
          System.Console.WriteLine("getQuoteOneWay was invoked.");
     }
     [WebMethod] [SoapRpcMethod]
     public float getQuote(String symbol) {
          if (symbol.ToUpper().Equals("DELAY")) Thread.Sleep(10000);
          return 88.88F;
     }
     [WebMethod] [SoapRpcMethod]
     public int asyncQuote(int delay) {
          Thread.Sleep(delay);
          return delay;
     }
     [WebMethod]
     public float getQuoteDOC(String symbol) {
          return 77.77F;
     }
}
```

Şekil 171. İç hizmet: `StockQuoteDotNet.asmx`

```
<%@ WebService Language="C#" Codebehind="SQDNNonInline.asmx.cs" Class="SQDNNonInline" %>
```

Şekil 172. Kod-arkasında: Tasarım `SQDNNonInline.asmx`

```
using System;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Web.Services.Description;
using System.Threading;

[WebService(Namespace = "http://stock.samples")]
public class SQDNNonInline : System.Web.Services.Protocols.SoapHttpClientProtocol
{

    [WebMethod]
    [SoapRpcMethod(OneWay = true)]
    public void getNonInlineQuoteOneWay(String symbol)
    {
        if (symbol.ToUpper().Equals("DELAY")) Thread.Sleep(5000);
        System.Console.WriteLine("getNonInlineQuoteOneWay was invoked.");
    }

    [WebMethod]
    [SoapRpcMethod]
    public float getNonInlineQuote(String symbol)
    {
        if (symbol.ToUpper().Equals("DELAY")) Thread.Sleep(10000);
        return 88.88F;
    }

    [WebMethod]
    [SoapRpcMethod]
    public int asyncNonInlineQuote(int delay)
    {
        Thread.Sleep(delay);
        return delay;
    }
    [WebMethod]
    public float getNonInlineQuoteDOC(String symbol)
    {
        return 77.77F;
    }
}
```

Şekil 173. Kod-arkasında: Uygulama: SQDNNonInline.asmx.cs

İlgili görevler

Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse
Hizmet sağlayıcınız olarak WebSphere MQ ' yı kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti geliştirin. 1.4eksenine yerleştirmeye yönelik bir hizmet oluşturmak için normal Web hizmeti geliştirme ortamınızı kullanın.

JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a JEE application server. This task is step 2 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol.

JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor

A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a JEE application server. This task is step 2 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol.

Başlamadan önce

EJB Web hizmetini yaratmak için Rational Application Developer olanağını kullanın. Rational Application Developer içindeki Web hizmeti sihirbazında, JMS üzerinde SOAP bağ tanımı için W3C aday önerisini kullanarak bir Web hizmeti yaratma seçeneği vardır. Rational Application Developer 7.54 gereklidir. The exercise used Rational Application Developer included in Rational Software Architect for WebSphere Software v7.5.5.1,

EJB, bu görevin bir parçası olarak Rational Application Developer uygulamasından WebSphere Application Server 'a konuşlandırılır. You must complete “Configuring WebSphere Application Server to use W3C SOAP over JMS” sayfa 966

Gerçekten görevde kullanılan WSDL ' yi yaratmak için öncelikle görevi tamamlamanız gerekir, “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922. Then you can either import the WSDL from the Dynamic Web project in the Eclipse Galileo workspace, or from the running HTTP Web service deployed to WASCE.

WebSphere Application Server might still be running as a result of doing “WebSphere Application Server kaynaklarını yapılandır” sayfa 967. Doğru değilse, bunu RAD ' deki Sunucular görünümünden başlatabilirsiniz.

Bu görev hakkında

In this task, you redeploy the `StockQuoteAxis` service from running as a JAX-RPC Axis service run by the **`SimpleJavaListener`** using WebSphere MQ transport for SOAP, to being a JAX-WS service running in WebSphere Application server using the W3C SOAP over JMS protocol.

There are two parts to migrating the service from the **`SimpleJavaListener`** to WebSphere Application Server:

1. Rational Application Developer olanağında Top-down EJB Web hizmeti sihirbazını kullanarak, hizmet için WSDL ' den Web hizmeti arabirimi oluşturun.
2. Implementing the service by importing the WebSphere MQ SOAP sample `StockQuoteAxis.java`.

Diğer bir yaklaşım, hizmeti `StockQuoteAxis.java`' den yukarıya doğru oluşturmak için olurdu. Ancak, yeni düzeye geçirilen hizmete ilişkin arabirimin aynı olduğundan emin olmak için, aynı WSDL ' yi kullandığı için yukarıdan aşağı yaklaşım daha iyi olur.

JMS desteği EJB taşıyıcısının bir parçası olduğu için, Web hizmeti değil, EJB taşıyıcısı için Web hizmeti geliştirilir.

Yordam

1. Rational Application Developer' ı başlatın ve WebSphere Application Server sunucusunun çalıştığını doğrulayın.
   a) Rational Application Developer olanağını yeni bir çalışma alanında başlatın.
   b) Java EE perspektifini açın.
   c) **Servers** (Sunucular) etiketini açın ve WebSphere Application Server sunucusunun çalıştığını doğrulayın.
      - Görünümde WebSphere Application Server v7.0 yoksa, görünüm > **Yeni** > **Sunucu**öğelerini sağ tıklatın. Bir WebSphere Application Server v7.0 örneği yaratmak için sihirbazdaki seçimleri izleyin.
      - Sunucu varsa, ancak başlatılmamışsa, başlatmak için ok başını tıklatın.
      - Özellikleri doğrulamak ve sunucu günlüklerine hızlı erişim elde etmek için **WebSphere Application Server v7.0 at localhost** > **Özellikler** > **WebSphere Application Server**seçeneğini sağ tıklatın.
      - Sunucuyu yönetmek için bir dış tarayıcı kullanın ve URL ' yi (`http://localhost:9061/ibm/console/unsecureLogon.jsp`) açın ya da **WebSphere Application Server v7.0 at localhost** > **Denetim konsolunu çalıştır**seçeneğini sağ tıklatın.
      - Varsayılan ayar otomatik olarak yayınlamasıdır. Birçok kişi, güncellemeleri sunucuya el ile devreye almayı tercih eder. Double-click **WebSphere Application Server v7.0 at localhost**, and expand the **Yayınlanıyor** twisty in the**Genel Bakış** window. **Hiçbir zaman otomatik olarak yayınlama**öğesini tıklatın.
      - Değiştirmek isteyebileceğiniz başka bir varsayılan değer ise, **Genel Bakış** penceresinde **Sunucuyu sona erdirme çalışma ortamının sona erdirilmesinde sonlandır** onay kutusunun işaretini kaldırın.

2. JEE projelerini yarat

   EAR (enterprise application project; bir EAR) ve bir Enterprise Java Bean (EJB) projesi yaratmalısınız.

   a) **Dosya** > **Yeni** > **Kurumsal Uygulama Projesi**. Name the project `W3CJMSEAR` > **Son**.

      The defaults must identify WebSphere Application Server v7.0 as the target runtime, and EAR version 5.0. Varsayılan yapılanış seçilmeli.

   b) **Dosya** > **Yeni** > **EJB Projesi**. `W3CJMSEJB`projesinin adını yazın. **EAR Projesi Adı** > **İleri**olarak `W3CEARJMS` seçeneğini belirleyin.

      Varsayılan EJB birimi sürümü 3.0 ' dır ve varsayılan yapılanış yeniden kullanılır.

   c) **Create an EJB Client JAR module** onay kutusunu temizleyin > **Finish**(Son).

3. Generate and deploy the EJB Web service from the `StockQuoteAxis` WSDL.

   a) **Çalıştır** > **Web Hizmetleri Gezgini 'ni başlat**.

   b) Select the WSDL page using the icons in the**Web Hizmetleri Gezgini** window > click **WSDL ana dosyası** in the Navigator.

   c) In the **Eylemler** window, type in or browse to the WSDL URL for `StockQuoteAxis.wsdl`.

      If you have WASCE running with `StockQuoteAxis` deployed as an HTTP service, the URL is:

      ```
      http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis?wsdl
      ```

      WSDL ' nin dosya sisteminde varsa, URL adresi şöyle olabilir:

      ```
      File:\Dirpath\StockQuoteAxis\WebContent\wsdl\StockQuoteAxis.wsdl
      ```

   d) Navigator ağacındaki içe aktarılan URL ' yi içeren satırı tıklatın.

      Bu, Web Hizmetleri Gezgini 'ne (içe) aktardığınız ilk WSDL ise, **WSDL Ana**' nın hemen altında olan satırdır.

   e) **Eylemler** penceresinde, **Web Hizmeti Sihirbazını Başlat** > **Web Hizmeti İskeleti** > **Git**öğelerini tıklatın.

   f) Web Hizmeti sihirbazında **Top down EJB Web Service**öğesini seçin.

      Select or verify the Configuration using information from Çizelge 141 sayfa 931 Check **Uyarı görüntülemeden dosyaların üzerine yaz** > **Sonraki**.

      Çizelge 141. EJB Web hizmeti yapılanışının üst kısmında

Alan	**Değer**
Sunucu	`WebSphere Application Server v7.0`
Web hizmeti çalıştırma zamanı	`IBM WebSphere JAX-WS`
Hizmet projesi	`W3CJMSEJB`
Hizmet EAR projesi	`W3CJMSEAR`
Yapılandırma:	`No client generation`

   g) **WebSphere JAX-WS EJB Top Down Web Hizmeti oluşturmaya ilişkin seçenekleri belirtin**başlıklı sayfada **JMS bağ tanımına geç**kutusuna onay imi girin. Ayrıca, **Sarıcı Stilini Etkinleştir**, **WSDL ' yi projeye kopyala** ve **Web Hizmeti Konuşlandırma Tanımlayıcısı Oluştur** > **Sonraki**öğelerini de denetleyin.

   h) On the page titled, **WebSphere JAX-WS JMS Bağ Tanımı Yapılanışı**, check **SOAP/JMS birlikte çalışabilirlik protokolünü kullan** and provide values from Çizelge 142 sayfa 932, leaving other fields blank > **Sonraki**.

Çizelge 142. WebSphere JAX-WS JMS Bağ Tanımı Yapılanışı	
Alan	**Değer**
JMS hedefi	queue
Hedef JNDI adı:	requestaxis
JMS bağlantı üreticisi	qm1
Yanıt Adı	W3CJMSEAR
Yapılandırma:	replyaxis

a) **WebSphere JAX-WS Yöneltici Projesi Yapılanışı**başlıklı sayfada, **ActivationSpec JNDI adı** alanında qm1as yazın > **İleri**.

RAD, projeyi oluşturmak ve konuşlandırmak için bir dakikanın yaklaşık 30 saniyeyi alır.

b) **Web Hizmeti Yayını** sayfasındaki > **Son**seçeneklerini dikkate almayın.

4. Oluşturulan WSDL ' yi denetleyin.

Hizmete özgü WSDL ' nin yaratılıp projeye saklanabilmesini istediniz.

a) Enterprise Explorer gezgininde, **W3CJMSEJB** > **ejbmodule** > **META-INF** > **wsdl**klasörünü açın. WSDL düzenleyicide açmak için StockQuoteAxis.wsdl simgesini çift tıklatın.

Bağlamayı inceleyin; JMS Url adresini görürsünüz:

```
jms:jndi:requestaxis?jndiConnectionFactoryName=qm1&targetService=StockQuoteAxis
```

5. İsteğe bağlı adım: EJB ' yi JAX-WS kullanarak HTTP üzerinden SOAP 'a bağlayın.

EJB ' ye iki bağ tanımı sağlamak, istemcilere Web hizmetini çağırmak için SOAP bağ tanımları seçmesini sağlar. Ayrıca, HTTP kullanarak WSDL ' yi almak için Web sunucusunu sorgulamak için kullanılacak araçları da sağlar.

Bir EJB ' yi HTTP üzerinden SOAP 'a bağlama adımları, görevin bir parçası olarak içerilmiyor.

6. Implement and redeploy StockQuoteAxis using the sample StockQuoteAxis.java

a) In the Enterprise Explorer navigator, open the folder **W3CJMSEJB** > **Hizmetler** Double click StockQuoteAxisService to open the implementation class in a Java editor.

b) StockQuoteAxis.java örnek programını *WebSphere MQ Installation directory*\tools\soap\samples\java\server klasöründe açın > Tüm yöntemleri seçin, ancak sınıf adını değil > **Kopyala**' yı seçin.

c) In StockQuoteAxisSoapBindingImpl.java select all the methods, but not the class name, and paste in the methods from StockQuoteAxis.java.

d) Add a print statement to output to the WebSphere Application Server console when the service is called.
getQuote(Dizgi simgesi) yöntemini değiştirin:

```
public float getQuote(String symbol) {
    System.out.println("StockQuoteAxisSoapBindingImpl called with symbol: "
                       + symbol);
    return ((float) 55.25);
```

e) İçe aktarma öğelerini düzeltin: **Kaynak** > **İçe aktarmayı düzenle** > **Kaydet**.

f) Arabirimle eşleşmeyen somutlama nedeniyle üç hatayı düzeltin.

The errors are due to three of the methods in StockQuoteAxis.java throwing exceptions, and the WSDL for the service not containing any fault messages. Sorun, yöntem imzaları ile yöntem Web hizmeti ek açıklamaları arasında uyuşmazlık olarak tanımlanıyor.

Yöntemlere @WebFault ile ek açıklama ekleyin ve WSDL ' yi yeniden oluşturun ya da arabirimi değiştirmeden tutun ve kural dışı durumları kaldırın.

Arabirimi aynı tutmak için, yöntem imzalarından üç `throws exception` ' yi kaldırın > **Kaydet**.

Sonraki adım

“JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma” sayfa 977

İlgili görevler

Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse
Hizmet sağlayıcınız olarak WebSphere MQ ' yı kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti geliştirin. 1.4eksenine yerleştirmeye yönelik bir hizmet oluşturmak için normal Web hizmeti geliştirme ortamınızı kullanın.

Developing a .NET 1 or 2 service for WebSphere MQ transport for SOAP using Microsoft Visual Studio 2008
Develop the SampleStockQuote Web service for .NET 1 or .NET 2 using Microsoft Visual Studio 2008

SOAP için WebSphere MQ iletimi için WebSphere MQ Web hizmeti istemcilerinin geliştirilmesi

SOAP için WebSphere MQ iletimi ile kullanmak üzere Web hizmeti istemcileri geliştirmek için normal geliştirme ortamınızı kullanın.

Başlamadan önce

Hizmeti yaratın. “SOAP için WebSphere MQ iletimi için Web hizmetleri geliştirilmesi” sayfa 921içindeki örneklerden birini kullanabilirsiniz.

İstemcileri nasıl geliştireceğini, konuşlandıracağını ve kullanacağını ve istemci nesli için WSDL ' yi nereden alacağına ilişkin seçenekleri belirleyin.

SOAP için WebSphere MQ iletimi için istemcilerin ve hizmetlerin geliştirmesine ilişkin yaklaşımınıza karar verin.

İki yaklaşım var.

1. Standart geliştirme araçlarını kullanın, bir HTTP hizmeti ve istemcisi geliştirin ve daha sonra, SOAP için WebSphere MQ iletimi URL 'sini kullanın.
2. SOAP için WebSphere MQ iletimi ile sağlanan araçları ve örnekleri kullanın.

HTTP rotasını kullanıyorsanız, hizmeti bir HTTP sunucusunda çalıştırabilir ve bunu SOAP için WebSphere MQ iletimi kullanarak çalıştırabilirsiniz. Bunu SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için, uygun WebSphere MQ dinleyicisini SOAP için yapılandırın ve hizmeti çalıştırmak için yolları ve konuşlandırma tanımlayıcılarını ayarlayın. The tools provided by WebSphere MQ transport for SOAP do the configuration for you. Diğer bir seçenek olarak, ortamı dinleyicilerin çalıştırılacağı şekilde yapılandırabilirsiniz.

SOAP için WebSphere MQ iletimi ile sağlanan araçlar, başlangıç ve iletim olanağının nasıl konuşlandırılacağı öğrenilmesinde yararlı olur. Üretim çalışmaları için, standart araçların kullanılmasında ve aynı hizmetin farklı SOAP aktarımlarıyla erişilebilir bir şekilde devreye alınması ile ilgili avantajlar vardır.

Geliştirilecek istemci tipine karar verin

Hangi Web hizmeti istemcisinin geliştirileceği konusunda karar vermelisiniz. Bu seçenek, hizmet arabirimini ve hizmetin adresini bilmenize bağlı olarak değişir.

Arabirimin bilinmesi durumunda, hizmet arabiriminden yetkili istemci sınıfları oluşturmak için Axis ya da .NET araçlarını kullanın. Yetkili istemci sınıfları, hizmeti çağırmak için istemci yazılmasını kolaylaştırır. Hizmetin yeri, istemciyi geliştirdiğinizde biliniyorsa, statik yetkili sunucu arabirimini kullanın. Hizmetin yeri değişirse (örneğin, hizmet bir üretim sunucusunda yeniden konuşlandırıldıysa), daha sonra dinamik yetkili sunucu arabirimini kullanın.

Hizmet arabirimi bir istemci geliştirdiğiniz sırada bilinmiyorsa, Axis için bir Dinamik Başlatma Arabirimi (DII) istemcisi oluşturabilirsinizAxisAxis 1.4. DII istemcisi, herhangi bir hizmeti çağırmak için soysal

bir arabirim kullanır. Parametreleri belirli bir hizmete doğru bir şekilde geçirmek için, belirli hizmet arabirimini programsal olarak oluşturmanız gerekir. Arabirimi istemcide programlı olarak ya da hizmet için WSDL ' yi istemciye yükleyerek oluşturun. Axis2' de bir Dağıtım istemcisi yaratabilirsiniz. Dağıtım istemcisi, istemci isteğini açıklamak için bir belge modeli kullanır, ancak bir DII istemcisi bir çağrı modelini kullanır. Her ikisi de isteği dinamik olarak oluşturma üzerinde çalışır.

Hizmet için WSDL ' yi edinin

Hizmet arabiriminin programlı olarak oluşturulması dışında, bir Web hizmeti istemcisi yaratmak için öncelikle hizmet WSDL ' yi edinmeniz gerekir. Hizmet WSDL, üç farklı kaynaktan edinilebilir:

1. Doğrudan Web hizmeti uygulamasından, **java2wsdl** (Axis) ya da **disco** (.NET) gibi bir araç kullanılarak.
2. URL kullanılarak Web hizmeti sorgulanarak: *Web service http url*?wsdl.
3. Bir dosya sisteminde ya da UDDI ya da WebSphere Service Registry and Repository gibi bir kayıt dosyasından ya da dosya sisteminden.

Not: HTTP ' yi kullanarak hizmete erişilemiyorsa, WSDL sorgusu çalışmamaktadır. Hizmetin kendisi yalnızca, SOAP için WebSphere MQ iletimi kullanılarak kullanılabilir.

amqwdeployWMQService tarafından oluşturulan WSDL, **java2wsdl** ya da **disco**kullanılarak oluşturulan WSDL ile aynı değil. Oluşturulan WSDL, "Top Down" hizmetini yaratmak için başladığınız WSDL ' lerden de farklıdır. Axis 'te, server-config.wsdd konuşlandırma tanımlayıcısı, bir istemci tarafından üretilen SOAP iletisini bir işlem ve hizmete eşler. **amqwdeployWMQService** , Eclipse' den farklı bir konuşlandırma tanımlayıcısı oluşturur.

İstemcileri oluşturmak için kullandığınız WSDL, hizmetin nasıl konuşlandırıldığı ile ilgili olarak değişir:

amqwdeployWMQServicekullanılarak konuşlandırıldı
amqwdeployWMQService tarafından oluşturulan WSDL ' yi kullanın. -w işaretini belirtin ve rpcLiteral WSDL ' yi seçin. Uyumluluk için, rpcEncoded WSDL ' yi seçebilirsiniz. rpcEncoded WSDL yalnızca .NET ve Axis 1.4 istemcileriyle çalışır.

SimpleJavaListenerkullanarak el ile devreye alma
Aşağıdaki WSDL dosyalarından birini kullanın:

1. Hizmeti tanımlamak için kullanılan WSDL ya da bir havuzda saklandı.
2. WSDL generated from the service by **java2wsdl** .
3. WSDL queried using the URL *Web service http url* ?wsdl, if available from an HTTP server. Hizmet tanımını doğrudan Eclipse' e aktarmak için Web Services Explorer gibi bir araç çalıştırabilirsiniz.

Hizmete ilişkin URI ' yi değiştirmeniz gerekebilir. HTTP hizmetinin adresinden, SOAP için WebSphere MQ iletimi URI 'sine ilişkin URI ' yı değiştirin.

amqSOAPNETListenerkomutunu kullanarak el ile devreye alma.
Aşağıdaki WSDL dosyalarından birini kullanın:

1. Hizmeti tanımlamak için kullanılan WSDL ya da bir havuzda saklandı.
2. .NET hizmeti sınıfından (.asmx) alınan WSDL. **disco**komutunu kullanma
3. WSDL queried using the URL *Web service http url* ?wsdl, if available. Hizmet tanımını doğrudan Eclipse' e aktarmak için Web Services Explorer gibi bir araç çalıştırabilirsiniz.
4. **amqswsdl** çalıştırılarak .NET hizmeti sınıfına (.asmx) karşı WSDL elde edildi.

Hizmete ilişkin URI ' yi değiştirmeniz gerekebilir. HTTP hizmetinin adresinden, SOAP için WebSphere MQ iletimi URI 'sine ilişkin URI ' yı değiştirin.

Windows Communication Foundation 'a konuşlandırıldı
Obtain the service WSDL by using the URL *Web service http url*?wsdl. Hizmet, hizmet tanımının bir parçası olarak serviceMetaVeri davranış yapılandırmasıyla tanımlanmalıdır.

Farklı bir sunucu platformuna konuşlandırma.
Doğru hizmet WSDL ' nin nasıl elde edileceği ile ilgili platform ile sağlanan kılavuzları izleyin.

Bu görev hakkında

Standart geliştirme araçlarını kullanarak müşterileri geliştirin. Aşağıdaki görevler, .NET 1 ve 2, Axis 1.4 (JAX-RPC) ve Axis2 (JAX-WS) için istemcilerin nasıl oluşturulacağı gösterilmektedir. Windows Communication Foundation için, ilgili görev bağlantılarına bakın.

Developing a JAX-RPC client for WebSphere transport for SOAP using Eclipse

SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti istemcisi geliştirin.

Başlamadan önce

Kullanılabilir bir hizmete sahip olmanız gerekir. If you are following the task as a practical exercise, use the workspace and service you created in the task, “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922. Eclipse içinde çalışan bir uygulama sunucusunun, Axis 1.4 Web hizmetlerini destekleyen bir uygulama sunucusu olması gerekir. Bu görevde, serbestçe kullanılabilir WebSphere Application Server Community Edition Sürüm 2.1.4' ü kullanırız. It is configured as part of the task “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922. Daha küçük bir açık kaynak uygulama sunucusu olan Tomcat 6 'yı da kullanabilirsiniz.

Bu görev hakkında

The task shows the development of three types of client for the sample StockQuoteAxis service using Eclipse running on Pencereler. İstemciler, istemci yetkili sunucusu ve bir DII istemcisi kullanılarak geliştirilmiş bir statik ve dinamik bir istemcidir.

WSDL ' den istemci yetkili sunucularını oluşturmak için iki alternatif yaklaşım gösterilmektedir:

1. **amqwdeployWMQService** kullanarak istemci yetkili sunucuları oluşturuluyor.
2. WSDL 'yi Eclipse' e (içe) aktarın ve istemci yetkili sunucuları oluşturmak için Web hizmeti sihirbazını kullanın.

Yordam

1. Start the Eclipse IDE for Java EE developers.
2. `StockQuoteAxisClient`adlı bir Java projesi yaratın:
   a) Java perspektifi > **Dosya** > **Yeni** > **Java Projesi**' ne geçin. **Java Project sayfasının yaratılması** tipi **Project name** alanında `StockQuoteAxisEclipseClient`. Yürütme ortamının **J2SE1-1.4** ya da **J2SE-1.5** > **Next** (İleri) olduğundan emin olun.
   b) **Java Settings** (Java Ayarları) sayfasında **Libraries** (Kitaplıklar) sekmesini seçin > **Add External JARs ...**(Dış JAR Ekle
   c) *MQ_INSTALLATION_PATH*`/java/lib` 'a göz atın ve tüm `.jar` dosyalarını > **Aç**' ı seçin.
   MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu dizindir.
   d) *MQ_INSTALLATION_PATH*`/java/lib/soap` 'a göz atın ve tüm `.jar` dosyalarını > **Aç**' ı seçin. `axis.jar` ' ı WebSphere MQ kuruluş ortamından bu dizine kurmuş olmanız gerekir.
   MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu dizindir.
   e) **Kitaplık** sekmesi artık istemciyi oluşturmak için gerekli olan tüm `.jar` dosyalarına başvurur > **Son**.
3. Örnek StockQuoteAxis Web hizmeti için Eclipse ' ta yetkili sunucular oluşturmak için bu iki yaklaşımın birini izleyin:
   - Generate the client proxies using **amqwdeployWMQService** .
     a. Bir kuyruk yöneticisi yaratın. Görev için, varsayılan kuyruk yöneticisi olarak `QM1` yaratın.
     b. Bir çalışma dizini oluşturun, `samples`. `StockQuoteAxis.java` örnek programını `samples/soap/server`içine kopyalayın.
     c. Modify `amqwsetcp.cmd` in *MQ_INSTALLATION_PATH*`/bin` to include the current directory in the classpath. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir.
     d. `samples` ' ta bir komut penceresi açın ve değiştirilen **amqwsetcp** komutunu çalıştırın.

e. Komutu çalıştırarak StockQuoteAxis hizmeti için WSDL yaratın.

```
amqwdeployWMQService -f soap/server/StockQuoteAxis.java -c genAxisWsdl
-u "jms:/queue?destination=REQUESTAXIS
&initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=(connectQueueManager(QM1)binding(auto))"
```

Unutmayın: Java komutları kullanırken "`.`" ya da "`\`" yerine "`/`"kullanın.

İpucu: Oluşturulan yetkili sunucuları Eclipse'e (içe) aktarmak yerine, oluşturulan WSDL' yi `.samples/generated`' den içe aktarabilirsiniz. Sonuçta ortaya çıkan yetkili sunucular iki şekilde farklılık gösterir:

i) Paket adları farklı; bu durumda yeniden canlanabilirsiniz.

ii) The Eclipse generated proxies include an additional helper class, `StockQuoteAxisProxy.java`

f. Create the client proxies for the StockQuoteAxis service by running the command:

```
amqwdeployWMQService -f soap/server/StockQuoteAxis.java -c genProxiestoAxis
-u "jms:/queue?destination=REQUESTAXIS
&initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=(connectQueueManager(QM1)binding(auto))"
```

g. İstemci yetkili sunucularını StockQuoteAxisClientiçine aktarın:

i) Right click **StockQuoteAxisClient\src** > Select **Dosya Sistemi** > **Sonraki** > **Göz At ...** > find the folder `.\samples\generated\client\remote\soap\server` > **Tamam**.

ii) **İçe Aktar** sayfasında > **Son**' da **sunucu** seçeneğini işaretleyin.

h. Paket adını `soap.server`olarak yeniden düzenleyin.

i) İstemci yetkili sunucularını içeren paketi farenin sağ düğmesiyle tıklatın > **Yeniden Düzenle** > **Yeniden Adlandır**. **`New name`**: `soap.server` > seçilen varsayılanları diğer seçenekler için bırakın > **Tamam** . Tüm hatalar düzeltiliyor.

- Generate the client proxies using Eclipse.

  Hizmet için WSDL ' yi edinmenin yollarından birini seçiniz. Bu örnekte, hizmet WebSphere Application Server Community Edition 'a konuşlandırılmıştır ve WSDL' yi Web sunucusundan edinmeniz gerekir. Konuşlandırma, “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922görevinde açıklanmıştır.

  a. Eclipse' de Web perspektifine geçin ve WebSphere Application Server Community Edition v2.1 Server 'ın çalışır durumda olduğunu ve StockQuoteekseninin konuşlandırıldığını ve uyumlulaştırıldığını doğrulayın.

  b. WSDL ' yi Web Hizmetleri Gezgini 'ne aktar:

  i) Eylem çubuğunda **Web Hizmetleri Gezgini** simgesini tıklatın ya da **Çalıştır** > **Web Hizmetleri Gezgini 'ni Başlat**seçeneklerini tıklatın.

  ii) WSDL sayfasına geçmek için Web Hizmetleri Gezgini 'nde WSDL sayfası simgesini tıklatın.

  iii) Web Hizmetleri Gezgini 'nin Navigator penceresinde **WSDL Ana Sayfası** öğesini tıklatın.

  iv) Web hizmetinin URL 'sini yazın ve ardından `?WSDL` yazın. The URL for StockQuoteAxis, deployed in the task “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922, is:

  ```
  http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis?wsdl
  ```

  c. İstemci yetkili sunucularının oluşturulması:

  i) Web Services Explorer gezgininde, **http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis?wsdl** seçeneğini tıklatın.

  ii) **Eylemler** penceresinde, **Web Hizmeti Sihirbazını Başlat** > **Web Hizmeti İstemcisi** ' nin seçilmesini bırak > **Git**seçeneğini tıklatın.

iii) Sihirbazın ilk sayfasında, yapılandırmadaki **İstemci** proje bağlantısını tıklatın > **StockQuoteAxisClient** istemci projesi > **Tamam**' ı seçin.

İpucu: Sihirbaz penceresi odaklanmayı kaybedebilir. Onu el ile odaklamak için geri getirmen gerekiyor.

iv) Bir JAX-RPC istemcisi oluşturmak için Web hizmeti yürütme ortamının Apache ekseninin olması gerekir.

v) **Bitir**'i tıklatın.

vi) Hizmetin statik URL 'sini, StockQuoteAxis hizmetine ilişkin SOAP adresi için WebSphere MQ iletimi için işaret edecek şekilde değiştirin. İstemciyi bir HTTP sunucusuyla test etinceye kadar bu adımı atlamayı seçebilirsiniz.

a) Open `StockQuoteAxisServiceLocator.java` and find the declaration for `StockQuoteAxis_address`.

b) URL ' yi değiştir

```
"jms:/queue?destination=REQUESTAXIS
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
&amp;connectionFactory=(connectQueueManager(QM1)binding(auto))"
```

İpucu: Eclipse automatically transforms & to `&`, and the reverse, when you copy and paste strings into `.java` code.

d. Her biri ana yöntemle olmak üzere üç Java istemcisi sınıfı yaratır:

i) Paket yarat. **StockQuoteAxisClient/src** > **Yeni Paket**' i sağ tıklatın. Adını `soap.client` > **Son**olarak adlandır.

ii) **soap.client** > **New** > **Class**öğelerini seçin. `SQAStaticClient` sınıfını adlandır > Denetle **public static void main (string [] args)** > Son

iii) Repeat the procedure to create `SQADynamicClient.java` and `SQADIIClient.java`

e. İstemci kodunu yazın.

Şekil 177 sayfa 941 - Şekil 181 sayfa 942 arasında, istemci kodunun üç stiline ilişkin örnekler sağlanır. Örnekler, bir HTTP sunucusuna konuşlandırılan StockQuoteAxis hizmetini kullanarak istemciyi sınamak için bir HTTP URL 'si kullanır. SOAP için WebSphere MQ iletimi kullanılarak konuşlandırılan StockQuoteAxis hizmetine karşı istemciyi çalıştırmak için URL adresini şu şekilde değiştirin:

```
"jms:/queue?destination=REQUESTAXIS
connectionFactory=(connectQueueManager(QM1)binding(auto))
initialContextFactory=com.ibm.mq.jms.Nojndi
targetService=soap.server.StockQuoteAxis.java
replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE"
```

– Şekil 177 sayfa 941 ve Şekil 179 sayfa 941 , kodlamayı biraz daha kolay kılan ek StockQuoteAxisproxy yardımcı sınıfına sahip olan Eclipsetarafından oluşturulan yetkili sunucuyu kullanır.

– Şekil 178 sayfa 941 ve Şekil 180 sayfa 942 , **amqwdeployWMQService** tarafından oluşturulan yetkili sunucuyu kullanır.

– Şekil 181 sayfa 942 yetkili sunucu sınıfı kullanmaz.

İstemcilerin her biri, SOAP için WebSphere MQ iletimi ile bağlantı oluşturmak için `com.ibm.mq.soap.Register.extension()` ' i arar. Uzantı, istemci konuşlandırma tanımlayıcısında kayıtlı. Client deployment to Axis 1.4 is described in “Deploying a Web service client to Axis 1.4 to use IBM WebSphere MQ transport for SOAP” sayfa 972.

f. SOAP isteğini, çalışma alanında yapılandırılan WebSphere Application Server Community Edition sunucusu tarafından barındırılan StockQuoteeksenine göndererek istemcileri çalıştırın.

i) Sunucunun çalıştığından, StockQuoteekseninin konuşlandırıldığını ve uyumlulaştırıldığını doğrulayın.

ii) Test etmek istediğiniz istemciyi seçin ya da açın > İşlem çubuğunda **Çalıştır** düğmesini tıklatın. Diğer bir seçenek olarak, yeşil Çalıştır simgesini ya da sekiz fareyi tıklatarak gezginde istemciyi tıklatın > **Farklı Çalıştır** > **Yapılandırmaları Çalıştır ...**seçeneğini tıklatın. İstemciyi çalıştırmak için gerek duyduğunuz parametreleri yapılandırın.

g. SOAP için WebSphere MQ iletimi kullanarak istemciyi çalıştırın.

Yordam, hizmeti konuşlandırmak için **amqwdeployWMQService** kullanır ve yalnızca **amqwdeployWMQService** tarafından oluşturulan WSDL ya da yetkili sunucuları kullanan istemciyle birlikte çalışır. Özgün WSDL ' yi ya da Eclipse tarafından oluşturulan yetkili sunucuları kullanarak istemciyi çalıştırmak için, hizmeti Eclipsetarafından oluşturulan konuşlandırma tanımlayıcısıyla devreye alın. Manually start **SimpleJavaListener** using the service port binding name as the `targetServiceAd` .

i) Hizmeti WebSphere MQ Simple Java SOAP dinleyicisine konuşlandırmak için "Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService" sayfa 960 içindeki yönergeleri izleyin. Hizmet devreye alma işlemi yalnızca **amqwdeployWMQService**tarafından oluşturulan WSDL ya da istemci yetkili sunucularını kullanan istemci için çalışır.

ii) Bir komut penceresinde, istemci konuşlandırma tanımlayıcısı dosyasını (`client-deploy.wsdd`) yaratmak için **amqwclientconfig** komutunu çalıştırın.

iii) `client-deploy.wsdd` dosyasını, SOAP için WebSphere MQ iletimi kullanarak test etmek istediğiniz Java projesinin köküne aktarın.

a) Java projesini farenin sağ düğmesiyle tıklatın **StockQuoteAxisEclipseClient** > **Import** > **File system** > **Next** > **Browse ...**

b) `client-deploy.wsdd` > **Aç** > **İçe Aktar** sihirbaz sayfasındaki dizini seçin > sağ pencere gözünde `client-deploy.wsdd` dizinini seçin.

c) **dosyasını doğrulayın:** `StockQuoteAxisEclipseClient` girmiştir > **Son**.

iv) Bu projede bir Java uygulamasını çalıştırmak için kullanılan çalışma dizininin `StockQuoteAxisEclipseClient` dizini olduğunu doğrulayın:

Java projesini farenin sağ düğmesiyle tıklatın **StockQuoteAxisEclipseClient** > **Run as ....** > **Run Configurations ...** > **(x) = Bağımsız Değişkenler** etiketini seçin > Çalışma Dizininde **Varsayılan** radyo düğmesinin işaretlendiğini doğrulayın ve yol `StockQuoteAxisEclipseClient` olur. Diğer bir seçenek olarak, istemci yapılanışını içeren farklı bir yer ya da dosya seçmek için aşağıdaki seçeneklerden birini belirleyebilirsiniz:

- **Diğer:** > seçeneğini işaretleyin. Seçim seçeneğinizin bir dizin yolunu yazın.
- **VM bağımsız değişkenleri** penceresinde, `-Daxis.ClientConfigFile=`*`full path to client deployment descriptor file`*yazın

v) URL ' nin, SOAP için WebSphere MQ iletimi kullanılarak konuşlandırılan hizmeti gösterecek şekilde yapılandırıldığından emin olun. İstemciyi adım ii' de açıklandığı gibi çalıştırın.

İpucu: Tipik olarak, aşağıdaki hatalardan biriyle karşılaşabilirsiniz:

i) `Exception: No client transport named 'jms' found!.`

ii) Bir JMS bağlantısı hatası.

iii) `Exception: The AXIS engine could not find a target service to invoke! targetService is soap.server.StockQuoteAxis.java`

iv) `Exception: java.lang.InstantiationException: soap.server.StockQuoteAxis`

Açıklamalar:

i) `client-config.wsdd` is not found, or does include the line `<transport name="jms" pivot="java:com.ibm.mq.soap.transport.jms.WMQSender"/>` in `client-config.wsdd`.

ii) Bir oluşturma yolu sorunu olabilir; *MQ_INSTALLATION_PATH*/java/lib içindeki .jar dosyaları da dahil değildir. *MQ_INSTALLATION_PATH* , WebSphere MQ ' un kurulu olduğu dizindir.

iii) Service deployment problem, either with server-config-wsdd , or with parameters passed to **SimpleSoapListener** .

iv) Konuşlandırma tanımlayıcısı ile hizmetin somutlaması arasında uyuşmazlık var.

İstemciyi Eclipseiçinde çalıştırmakta zorlanırsanız, komut penceresi kullanmayı deneyin:

i) Çalışma alanı dizin ağacındaki StockQuoteAxisEclipseClient\bin dizinine geçin.

ii) Run **amqwsetcp** and **amqwclientconfig**

iii) java soap/client/SQAStaticClientkomutunu çalıştırın.

Örnek JAX-RPC Web hizmeti istemcileri

WebSphere MQ ile birlikte gönderilen örnek Java Web hizmeti istemcileri *MQ_INSTALLATION_PATH* \tools\soap\samples\java\clients' ta kurulur. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir.

SQAxis2Axis.java
SQAxis2Axis.java, Şekil 174 sayfa 939 , StockQuoteAxis hizmetini çağırmak için dinamik bir yetkili istemci istemcidir. Komut satırında bir URL sağlayarak, dinamik yetkili sunucu içinde derlenen hizmetin URL 'sini geçersiz kılabilirsiniz.

SQAxis2DotNet.java
SQAxis2DotNet.java, Şekil 175 sayfa 940, StockQuoteDotNet hizmetini çağırmak için dinamik bir yetkili istemci istemcidir. Komut satırında bir URL sağlayarak, dinamik yetkili sunucu içinde derlenen hizmetin URL 'sini geçersiz kılabilirsiniz.

WsdlClient.java
WsdlClient.java, Şekil 176 sayfa 940, StockQuoteDotNet ya da StockQuoteAxis hizmetini çağırmak için dinamik bir çağırma istemcidir. İstemci varsayılan olarak StockQuoteAxis hizmetini çağırır. Add the command-line option -D invoke the StockQuoteDotNet service and -w to provide a different port to the one in .\generated\StockQuoteDotNet_Wmq.wsdl

```
package soap.clients;
import java.net.URL;
import soap.server.*;
public class SQAxis2Axis {
   public static void main(String[] args) {
      com.ibm.mq.soap.Register.extension();
      try {
         StockQuoteAxisService locator = new StockQuoteAxisServiceLocator();
         StockQuoteAxis service = null;
         if (args.length == 0)
            service = locator.getSoapServerStockQuoteAxis_Wmq();
         else
            service = locator.getSoapServerStockQuoteAxis_Wmq(
                new java.net.URL(args[0]));
         System.out.println("Response: " + service.getQuote("XXX"));
      } catch (Exception e) {
         System.out.println("\n>>> EXCEPTION WHILE RUNNING ProxyClient DEMO <<<\n");
                           e.printStackTrace();
         System.exit(2);
      }
   }
```

Şekil 174. SQAxis2Axis.java

```
public class SQAxis2DotNet {
public static void main(String[] args) {
   com.ibm.mq.soap.Register.extension();
      try {
         StockQuoteDotNet locator = new StockQuoteDotNetLocator();
         StockQuoteDotNetSoap_PortType service = null;
         if (args.length == 0)
            service = locator.getStockQuoteDotNetSoap();
         else
            service = locator.getStockQuoteDotNetSoap(new java.net.URL(
                  args[0]));
         System.out.println("Response: " + service.getQuoteDOC("XXX"));
      } catch (Exception e) {
         System.out.println("\n>>> EXCEPTION WHILE RUNNING ProxyClient DEMO <<<\n");
                           e.printStackTrace();
         System.exit(2);
      }
   }
}
```

Şekil 175. SQAxis2DotNet.java

```
package soap.clients;
import com.ibm.mq.soap.*;
import org.apache.axis.utils.Options;
import java.net.URL;
import javax.xml.rpc.Call;
import javax.xml.rpc.Service;
import javax.xml.rpc.ServiceFactory;
import javax.xml.namespace.QName;
public class WsdlClient {
public static void main(String[] args) {
      String wsdlService, wsdlPort, namespace, wsdlSource, wsdlTargetURI, s;
      try {
         Register.extension();
         Options opts = new Options(args);
         if (opts.isFlagSet('D') != 0) {
            wsdlService = "StockQuoteDotNet";
            wsdlPort = "StockQuoteDotNetSoap";
            namespace = "http://stock.samples";
            wsdlSource = "file:generated/StockQuoteDotNet_Wmq.wsdl";
         } else {
            wsdlService = "StockQuoteAxisService";
            wsdlPort = "soap.server.StockQuoteAxis_Wmq";
            namespace = "soap.server.StockQuoteAxis_Wmq";
            wsdlSource = "file:generated/soap.server.StockQuoteAxis_Wmq.wsdl";
         }
         if (null != (s = (opts.isValueSet('w'))))
            wsdlPort = s;
         System.out.println("start WsdlClient demo, wsdl port " + wsdlPort
               + " resolving uri to ...");
         QName servQN = new QName(namespace, wsdlService);
         QName portQN = new QName(namespace, wsdlPort);
         Service service = ServiceFactory.newInstance().createService(
               new URL(wsdlSource), servQN);
         Call call = (Call) service.createCall(portQN, "getQuote");
         wsdlTargetURI = call.getTargetEndpointAddress().toString();
         System.out.println("  '" + wsdlTargetURI + "'");
         Object ret = call.invoke(new Object[] { "XXX" });
         System.out.println("Response: " + ret);
      } catch (Exception e) {
         System.out.println("\n>>> EXCEPTION WHILE RUNNING WsdlClient DEMO <<<\n");
         e.printStackTrace();
         System.exit(2);
      }
   }
}
```

Şekil 176. WsdlClient.java

Bu görevde kullanılan örnek istemciler şunlardır:

```
package soap.client;
import soap.server.StockQuoteAxisProxy;
public class SQAStaticClient {
  public static void main(String[] args) {
    try {
      com.ibm.mq.soap.Register.extension();
      StockQuoteAxisProxy sqa = new StockQuoteAxisProxy();
      System.out.println("Static client synchronous result is:"
          + sqa.getQuote("ibm"));
    } catch (Exception e) {
      System.out.println("Exception: " + e);
    }
  }
}
```

Şekil 177. Static client using Eclipse generated proxy

```
package soap.client;
import soap.server.StockQuoteAxis;
import soap.server.StockQuoteAxisService;
import soap.server.StockQuoteAxisServiceLocator;
public class SQAStaticClient {
  public static void main(String[] args) {
    try {
      com.ibm.mq.soap.Register.extension();
      StockQuoteAxisService locator = new StockQuoteAxisServiceLocator();
      StockQuoteAxis sqa = locator.getSoapServerStockQuoteAxis_Wmq();
      System.out.println("Static client synchronous result is: "
          + sqa.getQuote("ibm"));
    } catch (Exception e) {
      System.out.println("Exception: " + e);
    }
  }
}
```

Şekil 178. Static client using amqwdeployWMQService generated proxy

```
package soap.client;
import soap.server.StockQuoteAxisProxy;
public class SQADynamicClient {
  public static void main(String[] args) {
    try {
      com.ibm.mq.soap.Register.extension();
      StockQuoteAxisProxy sqa = new StockQuoteAxisProxy(
          "http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis");
      System.out.println("Dynamic client synchronous result is: "
          + sqa.getQuote("ibm"));
    } catch (Exception e) {
      System.out.println("Exception: " + e);
    }
  }
}
```

Şekil 179. Eclipse tarafından oluşturulan yetkili sunucu kullanılarak dinamik istemci

```
package soap.client;

import java.net.URL;
import soap.server.StockQuoteAxis;
import soap.server.StockQuoteAxisService;
import soap.server.StockQuoteAxisServiceLocator;
public class SQADynamicClient {
  public static void main(String[] args) {
    try {
      com.ibm.mq.soap.Register.extension();
      URL sqaURL = new URL(
          "http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis");
      StockQuoteAxisService locator = new StockQuoteAxisServiceLocator();
      StockQuoteAxis sqa = locator.getSoapServerStockQuoteAxis_Wmq(sqaURL);
      System.out.println("Dynamic client synchronous result is: "
          + sqa.getQuote("ibm"));
    } catch (Exception e) {
      System.out.println("Exception: " + e);
    }
  }
}
```

Şekil 180. amqwdeployWMQService tarafından oluşturulan yetkili sunucu kullanılarak devingen istemci

```
package soap.client;
import java.net.URL;
import javax.xml.namespace.QName;
import javax.xml.rpc.Call;
import javax.xml.rpc.Service;
import javax.xml.rpc.ServiceFactory;
public class SQADIIClient {
  public static void main(String[] args) {
    try {
      com.ibm.mq.soap.Register.extension();
      URL wsdl = new URL(
          "http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis?wsdl");
      Service SQAService = (ServiceFactory.newInstance()).createService(wsdl,
          new QName("http://server.soap", "StockQuoteAxisService"));
      Call SQACall = SQAService.createCall(new QName("http://server.soap",
          "StockQuoteAxis"), "getQuote");
      System.out.println("DII client synchronous result is "
          + SQACall.invoke(new Object[] { "ibm" }));
    } catch (Exception e) {
      System.out.println("Exception: " + e);
    }
  }
}
```

Şekil 181. DII istemcisi (Yetkili sunucu yok)

İlgili görevler

Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Axis2 Web hizmeti istemcisi geliştirin. SOAP için WebSphere MQ iletimiyle sağlanan örnek Axis2 istemcileri listelenir ve yetkili sunucular oluşturmak için kullanılan **wsimport** komutu kullanılır.

Developing a .NET 1 or 2 client for WebSphere transport for SOAP using Microsoft Visual Studio 2008
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir .NET 1 ya da 2 Web hizmeti istemcisi geliştirin.

Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse

SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Axis2 Web hizmeti istemcisi geliştirin. SOAP için WebSphere MQ iletimiyle sağlanan örnek Axis2 istemcileri listelenir ve yetkili sunucular oluşturmak için kullanılan **wsimport** komutu kullanılır.

Başlamadan önce

Obtain the Axis2 libraries, and configure a development and test environment to run the client.

Not: Eksen tarafından kullanılan sürümlerin ve yayınların adlandırılmasına karışıklığa neden olur. Tipik olarak, 1.4 ekseni JAX-RPC somutlamasını ve Axis2 JAX-WS somutlaması için kullanılır.

1.4 ekseni bir sürüm düzeyidir. İnternet 'te Eksen 1.4 için arama ararsanız, http://ws.apache.org/axis/' a götürülesiniz. The page contains a list of preceding versions of Axis (1.2, 1.3) and the April 22, 2006, final release of Axis 1.4. Daha sonra Axis 1.4yayın düzeyleri vardır, bu düzeltme hataları, ancak bunların tümü Axis 1.4olarak bilinirler. Bu, WebSphere MQile birlikte gönderilen bu hata düzeltme yayınlarından biridir. 1.4ekseni için, http://ws.apache.org/axis/ile verilenobtainablesürümünü değil, WebSphere MQ ile birlikte gönderilen `axis.jar` sürümünü kullanın.

The Axis website also refers to Axis 1.1 to refer to all the versions of what is more typically called Axis 1.4. Axis 1.2 is used to refer to what is typically called Axis2.

1.5 ekseni, 1.4ekseninin daha sonraki bir yayın düzeyi değil, bir Axis2 yayınıdır. If you search for Axis 1.5 you are directed to http://ws.apache.org/axis2/. https://ws.apache.org/axis2/download.cgi contains a list of release versions of Axis2, labeled 0.9 to 1.5.1 (and including, confusingly version 1.4). SOAP için WebSphere MQ iletimi ile kullanmak üzere Axis2 yayın düzeyi 1.4.1' dir. Axis2 1.4.1 dosyasını http://ws.apache.org/axis2/download/1_4_1/download.cgiolanağından yükleyin.

You can choose to generate proxies for the Web service clients for WebSphere MQ transport for SOAP using either **`wsimport`** or the tooling provided with an IDE. Eclipse IDE for Java EE Developer 3.5 SR1 uses **`wsdl2java`**.**`wsimport`** , Java 6 ile birlikte sağlanır. You can use Java 5 to run client proxies generated either with **`wsimport`** or **`wsdl2java`**.

SOAP için WebSphere MQ iletimi ile sağlanan örnek Web hizmeti Axis2 istemcileri, **`wsimport`**kullanılarak geliştirilmiştir; bkz. “Örnek Axis2 istemcileri” sayfa 948.

Aşağıdaki kısımda, Java EE geliştiricileri için Eclipse IDE ile paketlenen Web hizmetleri sihirbazı tarafından üretilen yetkili sunucuların nasıl oluşturulacağı ve kullanılacağı gösterilir. Örnek müşteriler, **`wsimport`**tarafından üretilen proxylerin nasıl kullanılacağını gösterir.

Web hizmetleri sihirbazını kullanmak için, Workbench 'e Axis2 ' yi destekleyen bir uygulama sunucusu eklemelisiniz. Bu adımlarda, çalışma ortamını kullanarak Axis2 'ı desteklemek için WASCE' nin nasıl yapılandırılacağı gösterilmektedir.

1. Configure the application server used in Eclipse IDE for Java EE Developers to support Axis2. In this example, configure the WASCE 2.1.4, application server, which is part of the workspace created in “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922.
   a. Sunucuyu yapılandırmak için çalışma alanı tercihlerini açın: Açık **Pencere** > **Tercihler**.
   b. Kurulu JRE 'nin `Java50`olup olmadığını denetleyin: **Kurulu JRE' ler**öğesini tıklatın.
   c. Sunucu olarak WASCE ' yi ekleyin: Tıklat **Sunucu** > **Yürütme ortamları** > **Ekle ...** > **IBM** > **WASCE v2.1** > İleri. JRE Java50 > WASCE kuruluş dizinine göz at > **Tamam** > **Son**olmalıdır. Web Geliştiricileri için Eclipse Java EE IDE 'si için WASCE eklentisini kurmuş olmanız gerekir.
   d. Add Axis2: Click **Web Hizmetleri** > **Axis2 Tercihleri**. **Axis2 Runtime** etiketinde > **Göz at ...** Birçok `Axis2` jar dosyasını (**Apply**) içeren dizini açın.
   e. WASCE 'yi Axis2: ile ilişkilendirin: **Web Hizmetleri** > **Sunucu ve Yürütme Ortamı**' yı tıklatın. **`Server`** altında, **IBM WASCE v2.1 Server**' ı seçin ve **`Web service runtime`**altında, **Apache Axis2** > **Uygula** > **Tamam**seçeneklerini belirleyin
   f. Sunucuyu başlatın: Web perspektifini açın ve Sunucular görünümünü açın. Sunucular görünümü > **Yeni** > **Sunucu**öğelerini sağ tıklatın. **IBM WASCE v2.1 Server** seçili ve yapılandırılmış > **Son**. Sunucuyu başlatın.
2. Check that you have deployed the StockQuoteAxis service to WASCE to run the Web service wizard.
3. To test the service with the WebSphere MQ transport for SOAP service, deploy the service to a WebSphere MQ transport for SOAP listener for Axis 1.4; see “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922.

Bu görev hakkında

The Eclipse IDE for Java EE Developers uses Java50 and the Web services wizard to generate the proxy classes for the service. Yetkili sınıflar, Java 6 ile birlikte sağlanan **`wsimport`** aracı tarafından yaratılan

sınıflarla farklıdır. Alternatif bir yaklaşım, yetkili sunucu sınıflarını **wsimport** kullanarak oluşturmada ve yarattığı paketleri, Web Geliştiricileri için Eclipse Java EE IDE ' nizde içe aktarmadır.

Eclipse IDE for Java EE Developers olanağında Web hizmetleri sihirbazı bir Web projesi içinde bir Web hizmeti istemcisi oluşturur. İstemciyi basit bir Java uygulaması olarak çalıştırabilirsiniz; bir uygulama sunucusu gerektirmez. Kodu bir Java projesine de aktarabilir ve oluşturma yolunu Axis2 JAR dosyalarını içerecek şekilde yapılandırabilirsiniz.

Yordam

1. Yeni bir kurum projesinde bir Web projesi yaratmanızı sağlar:
   a) Proje Gezgini 'nde hiçbir şey seçilmemesiyle > Beyaz alanı sağ tıklatın > **Yeni** > **Kurum Uygulaması Projesi** > Adı `StockQuoteAxis2EAR` > **Son**tıklatın. Reply No to the window giving you the option of opening the Java EE perspective.

      Varsayılan değerler, WASCE kullanacak şekilde ayarlanır.

   b) `StockQuoteAxis2EAR` > **Yeni** > **Dinamik Web Projesi**öğelerini sağ tıklatın. Name the project `StockQuoteAxis2WebClient` > Check the EAR membership box to add the project to **StockQuoteAxis2EAR**. Hedef yürütme ortamı olarak WASCE 2.1 seçilidir.

   c) **Yeni Dinamik Web Projesi** sayfasının Yapılanış kısmında > **Değiştir ...** > Axis2 Web hizmetleri projesi iç işlev kümesini denetleyin. **Dinamik Web Birimi 2.5**, **Java 6.0**ve **WASCE devreye alımı 1.2** önceden işaretlendi. > **Tamam** > **Son**. Reply No to the window giving you the option of opening the Java EE perspective.
2. Hizmete ilişkin WSDL ' yi çalışma alanına içe aktarın ve istemci yetkili sunucusunu oluşturun:

   Bu örnekte, WSDL belgesi HTTP hizmeti bağ tanımını içerir ve statik Web istemcisi yetkili sunucusu için hedef olur. Web hizmeti bağlayıcısındaki URL ' yi değiştirerek, istemci yetkili sunucusunu oluşturmadan önce SOAP URL 'sine ilişkin WebSphere MQ iletimi için kullanılacak bir iletim yolu seçin. Statik Web istemcisi yetkili sunucusu, daha sonra SOAP için WebSphere MQ iletimi için konuşlandırılan hizmettir.

   a) Web Hizmetleri Gezgini 'ni başlatın: eylem çubuğunda simgeyi kullanın ya da **Çalıştır** > **Web Hizmetleri Gezgini 'ni Başlat**.

   b) WSDL gezginini seçmek için, **Web Services Explorer** penceresinde WSDL simgesini tıklatın > Navigator penceresinde **WSDL Ana** öğesini tıklatın > StockQuoteAxis WSDL dosyasının URL adresini yazın > **Git**.
   Bu örnekte, WSDL ' yi doğrudan HTTP hizmetinden edinin: `http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis?wsdl`

   c) Navigator' de, Web hizmetinin URL 'sini içeren satırı tıklatın. In the **Eylemler** window, click **WSDL ' yi Workbench 'e Aktar** > Select a **StockQuoteAxis2WebClient** as the **Çalışma ortamı projesi** > Type the **WSDL dosyası adı**, `StockQuoteAxisHTTP.wsdl` > **Git**.

   d) **StockQuoteAxisHTTP.wsdl** > **Web Services** > **Generate Client**öğelerini sağ tıklatın. Sihirbazın Web hizmetleri sayfasına ilişkin yapılandırma bilgilerini denetleyin: Sunucu: IBM WASCE v2.1 Server, Web hizmeti yürütme ortamı: Apache Axis2, İstemci projesi: StockQuoteAxis2WebClient, Client EAR projesi: StockQuoteAxisEAR. Yapılandırmayı düzeltmek için, yanlış olan satırları tıklatın.

   e) Click **Sonraki** > verify the code generation settings > **Son**.

      Yeni bir paket (`soap.server`) oluşturulduğunu ve gereksinim duyduğunuz yetkili sunucuları içerdiğini fark edin.
3. Projeyi JMS iletimi olarak SOAP için WebSphere MQ iletimi çalıştırabilmek için yapılandırın.

   SOAP içinWebSphere MQ iletimi bir `transportSender`sağlar, ancak `transportReceiver`seçeneği yoktur. In other words, WebSphere MQ transport for SOAP supports Axis2 clients. Şu anda Axis2 hizmetlerini desteklememektedir.

   a) **StockQuoteAxis2WebClient** projesinde, `WebContent\WEB-INF\conf\axis2.xml` > **Birlikte aç ...** > **XML düzenleyicisi**öğelerini sağ tıklatın.

   b) Son `transportSender` (dosyanın sonuna doğru) için arama yapın ve açıklama satırı yapan JMS `transportSender` > satırını sağ tıklatın > **Önce ekle ...** > **transportSender**.

c) Sağ tıklatın **transportSender** > **Öznitelik Ekle** > **Ad** > **transportSender** > **Öznitelik Ekle** > **Sınıf**seçeneğini sağ tıklatın.

d) **Ad** > **Özniteliği Düzenle** > **Değer:** `jms`öğesini farenin sağ düğmesiyle tıklatın.

e) Sağ tıklatın **Sınıf** > **Özniteliği Düzenle** > **Değer:** `com.ibm.mq.axis2.transport.jms.WMQJMSTransportSender`. > Kaydet seçeneklerini tıklatın.

f) Oluşturma yoluna `com.ibm.mq.axis2.transport.jms.WMQJMSTransportSender` ekleyin: Sağ tıklatın **StockQuoteAxis2WebClient** > **Oluşturma Yolu** > **Oluşturma Yolu Yapılandır ...** > **Kitaplıklar** sekmesini tıklatın > **Dış JAR Ekle ...**. *MQ_INSTALLATION_PATH*`\java\lib` > **Tamam**'da tüm JAR' ları seçin.

MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu dizindir.

4. Zamanuyumlu bir statik istemci oluşturun, bunu HTTP kullanarak test edin ve daha sonra, SOAP için WebSphere MQ iletimi kullanarak statik istemciyi çalıştırmak için yetkili sunucuyu dönüştürün.

a) **Java Resources: src** > **New** > **Package** > Package adlı paketi farenin sağ düğmesiyle tıklatın > `soap.client` > Finish (Son) düğmesini tıklatın.

b) **soap.client** > **Yeni** > **Sınıf** > Sınıfın adını `SQA2StaticClient` > **Son**seçeneğini sağ tıklatın.

c) Sınıfı aşağıdaki kodla değiştirin ve **Save**(Kaydet) düğmesini tıklatın.

Şekil 182. `SQA2DynamicClient.java`

```
package soap.client;
import soap.server.StockQuoteAxisServiceStub;
import soap.server.StockQuoteAxisServiceStub.GetQuote;
public class SQA2StaticClient {
  public static void main(String[] args) {
    try {
      StockQuoteAxisServiceStub stub = new StockQuoteAxisServiceStub();
      GetQuote request = new GetQuote();
      request.setSymbol("ibm");
      System.out.println("Response is: "
          + (stub.getQuote(request)).getGetQuoteReturn());
    } catch (Exception e) {
      System.out.println("Exception: " + e.getMessage());
      e.printStackTrace();
    }
  }
}
```

5. WASCE ' ye yerleştirilen StockQuoteAxis hizmetiyle ve SOAP için WebSphere MQ iletimi ile istemciyi test edin.

a) Proje Gezgini 'nde, **SQA2StaticClient** > **Bu şekilde çalıştır ...** > **Java Uygulaması**öğelerini sağ tıklatın.

The result, `Response is 55.25`, appears in the Console view. Ayrıca, Konsol görünümünde WASCE konsol penceresini seçebilir ve WASCE sunucusundaki çıktıyı (`StockQuoteAxis called with parameter: ibm`) görebilirsiniz.

b) The proxy was built with the service address, `http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis`, and so the static client calls the service running on HTTP. Statik istemciyi, SOAP için WebSphere MQ iletimi kullanarak hizmeti çağırmak için değiştirebilirsiniz. The following instructions change the service address in `StockQuoteAxisServiceStub.java` without rebuilding the proxy, and configure the `SQA2StaticClient` runtime parameters to load `axis2.xml`. You configure `axis2.xml` configures Axis2 to use WebSphere MQ transport for SOAP.

c) Open `StockQuoteAxisServiceStub.java` > `http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis` ile ilgili iki geçiş sayısını değiştirin:

```
jms:/queue?destination=REQUESTAXIS@QM1
&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi
```

```
&targetService=StockQuoteAxis
&replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
```

d) SQA2StaticClient komutunu şimdi çalıştırırsanız, JMS için yapılandırılmış bir transportSender bulunamadığı için kural dışı durum oluşur.
Kural dışı durum:

```
Exception: null java.lang.NullPointerException at
soap.server.StockQuoteAxisServiceStub.getQuote(StockQuoteAxisServiceStub.java:547)
at soap.client.SQA2StaticClient.main(SQA2StaticClient.java:11)
```

e) Proje Gezgini 'nde **SQA2StaticClient** > **Bu şekilde çalıştır ...** > **Yapılandırmaları Çalıştır ...**öğelerini seçin. Switch to the **(x) = Bağımsız Değişkenler** tab, and in the **VM bağımsız değişkenleri** input area, type the path to the axis2.conf file > **Uygula** > **Çalıştır**.
VM bağımsız değişkeni: -Daxis2.xml=${workspace_loc:StockQuoteAxis2WebClient/WebContent/WEB-INF/conf}/axis2.xml. Ya da Axis2 yapılandırma dosyasına standart bir yol sağlayabilirsiniz.

f) SQA2StaticClient komutunu yeniden çalıştırın. Bu çalıştırmada, SOAP için WebSphere MQ iletimi olanağını kullanıyorsunuz. WASCE konsolunda yeni çıktı olup olmadığını denetleyerek bunu onaylayın. Open the console or command window that is associated with SimpleJavaListener, and the output there is StockQuoteAxis called with parameter: ibm.

6. HTTP için bir dinamik istemci ve SOAP için WebSphere MQ iletimi yaratın ve test edin.

a) **soap.client** > **Yeni** > **Sınıf** > Sınıfın adını SQA2DynamicClient > **Son**seçeneğini sağ tıklatın.

b) Sınıfı aşağıdaki kodla değiştirin ve **Save**(Kaydet) düğmesini tıklatın.

```
package soap.client;
import soap.server.StockQuoteAxisServiceStub;
import soap.server.StockQuoteAxisServiceStub.GetQuote;
public class SQA2DynamicClient {
  public static void main(String[] args) {
    try {
      StockQuoteAxisServiceStub stub = new StockQuoteAxisServiceStub(
          "http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis");
      GetQuote request = new GetQuote();
      request.setSymbol("ibm");
      System.out.println("HTTP Sync: "
          + (stub.getQuote(request)).getGetQuoteReturn());
      stub = new StockQuoteAxisServiceStub(
          "jms:/queue?destination=REQUESTAXIS@QM1"
          + "&connectionFactory=()&initialContextFactory=com.ibm.mq.jms.Nojndi"
          + "&targetService=StockQuoteAxis&replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE");
       System.out.println("JMS sync: "
          + (stub.getQuote(request)).getGetQuoteReturn());
     } catch (Exception e) {
       System.out.println("Exception: " + e.getMessage());
       e.printStackTrace();
     }
   }
 }
```

c) SQA2DynamicClient.javaiçin bir çalıştırma yapılandırması oluşturun ve yolu axis2.xml' a ekleyin:
-Daxis2.xml=${workspace_loc:StockQuoteAxis2WebClient/WebContent/WEB-INF/conf}/axis2.xml

d) SQA2DynamicClientkomutunu çalıştırın. SQA2DynamicClient, WASCE ve **SimpleJavaListener**için konsol çıkışını denetleyin.

7. Zamanuyumsuz bir istemci yaratın ve sonuç olarak geri çağırma işleyicisinde ve ana program iş parçacığıdaki sonuca erişin.

The asynchronous client proxies created by the Web service wizard for Eclipse Java EE IDE for Web Developers differ from the proxies created by **wsimport**. **wsimport** yetkili sunucuları, Future, Responseve AsyncHandler soysal tiplerini kullanır.

Web Geliştiricileri için Eclipse Java EE IDE ' nin Web hizmeti sihirbazı bir StockQuoteAxisServiceCallbackHandler soyut sınıfı yaratır.

StockQuoteAxisServiceCallbackHandler ' u genişletmeli ve bir geri çağırma işleyicisi oluşturmalısınız.

a) **soap.client** > **Yeni** > **Sınıf** > Sınıfın adını SQA2CallbackHandler > **Son**seçeneğini sağ tıklatın.

b) Sınıfı aşağıdaki kodla değiştirin.

```
package soap.client;
import soap.server.StockQuoteAxisServiceCallbackHandler;
import soap.server.StockQuoteAxisServiceStub.GetQuoteResponse;
public class SQA2CallbackHandler
    extends StockQuoteAxisServiceCallbackHandler {
  private boolean complete = false;
  SQA2CallbackHandler() {
    super();
    System.out.println("Callback constructor");
  }
  public void receiveResultgetQuote(GetQuoteResponse response) {
    System.out.println("Result in Callback " + response.getGetQuoteReturn());
    super.clientData = response;
    complete = true;
  }
  public boolean isComplete() {
    return complete;
  }
}
```

c) **soap.client** > **Yeni** > **Sınıf** > Sınıfın adını SQA2AsyncClient > **Son**seçeneğini sağ tıklatın.

d) Sınıfı aşağıdaki kodla değiştirin.

Şekil 183. SQA2AsyncClient.java

```
package soap.client;
import soap.server.StockQuoteAxisServiceStub;
import soap.server.StockQuoteAxisServiceStub.GetQuote;
import soap.server.StockQuoteAxisServiceStub.GetQuoteResponse;
import soap.server.StockQuoteAxisServiceCallbackHandler;
@SuppressWarnings("unused")
public class SQA2AsyncClient {
  public static void main(String[] args) {
    try {
      StockQuoteAxisServiceStub stub = new StockQuoteAxisServiceStub(
          "http://localhost:8080/StockQuoteAxis/services/StockQuoteAxis");
      GetQuote request = new GetQuote();
      request.setSymbol("ibm");
      System.out.println("HTTP Sync: "
          + (stub.getQuote(request)).getGetQuoteReturn());
      SQA2CallbackHandler callback = new SQA2CallbackHandler();
      stub.startgetQuote(request, callback);
      do {
        System.out.println("Waiting for HTTP callback");
        Thread.sleep(2000);
      } while (!callback.isComplete());
      System.out.println("HTTP poll: "
          + ((GetQuoteResponse) (callback.getClientData()))
                .getGetQuoteReturn());
      stub = new StockQuoteAxisServiceStub(
          "jms:/queue?destination=REQUESTAXIS@QM1"
          + "&connectionFactory=()&initialContextFactory=com.ibm.mq.jms.Nojndi"
          + "&targetService=StockQuoteAxis&replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE");
      System.out.println("JMS Sync: "
          + (stub.getQuote(request)).getGetQuoteReturn());
      callback = new SQA2CallbackHandler();
      stub.startgetQuote(request, callback);
      while (!callback.isComplete()) {
        System.out.println("Waiting for JMS callback");
        Thread.sleep(2000);
      }
      System.out.println("JMS poll: "
        + ((GetQuoteResponse) (callback.getClientData())).getGetQuoteReturn());
    } catch (Exception e) {
        System.out.println("Exception: " + e.getMessage());
        e.printStackTrace();
     }
```

```
    }
  }
```

Konsol çıkışı aşağıdaki gibidir:

```
HTTP Sync: 55.25
Callback constructor
Waiting for HTTP callback
Result in Callback 55.25
HTTP poll: 55.25
JMS Sync: 55.25
Callback constructor
Waiting for JMS callback
Result in Callback 55.25
JMS poll: 55.25
```

Örnek Axis2 istemcileri

Örnek yetkili sunucular, Java 6 ile paketlenmiş **wsimport** aracı kullanılarak oluşturulur. Altı örnek verilmiştir:

1. DynamicProxyClientSync.java
2. DynamicProxyClientAsyncPolling.java
3. DynamicProxyClientAsyncCallback.java
4. DispatchClientSync.java
5. DispatchClientAsyncPolling.java
6. DispatchClientAsyncCallback.java

Örnek StockQuoteAxis sunucusu için istemci örnekleri üretilir. Generate the WSDL with the **amqwdepoyWMQServer** command, specifying the -w switch to select `rpcLiteral` style. Örneklere ilişkin yetkili sunucular oluşturmak için aşağıdaki komutu kullanın:

```
wsimport soap.server.StockQuoteAxis_Wmq.wsdl -d generated -keep -p com.ibm.mq.axis2.samples
```

Şekil 184. DynamicProxyClientSync.java

```
package com.ibm.mq.axis2.samples;

import com.ibm.mq.axis2.samples.proxy.StockQuoteAxis;
import com.ibm.mq.axis2.samples.proxy.StockQuoteAxisService;

public class DynamicProxyClientSync {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DynamicProxyClientSync");

      System.out.println("Creating proxy instance for service StockQuoteAxisService");
      StockQuoteAxisService stub = new StockQuoteAxisService();
      StockQuoteAxis service = stub.getSoapServerStockQuoteAxisWmq();

      System.out.println("Invoking getQuoteOneWay OneWay operation synchronously...");
      service.getQuoteOneWay("48");
      System.out.println(" > getQuoteOneWay has returned");

      System.out.println("Invoking getQuote Request Reply operation synchronously...");
      float result = service.getQuote("48");
      System.out.println(" > getQuote has returned result of " + result);

      System.out.println("End of sample");
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
```

```
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }
}
```

Şekil 185. DynamicProxyClientAsyncPolling.java

```
package com.ibm.mq.axis2.samples;

import java.util.concurrent.CancellationException;

import javax.xml.ws.Response;

import com.ibm.mq.axis2.samples.proxy.StockQuoteAxis;
import com.ibm.mq.axis2.samples.proxy.StockQuoteAxisService;

public class DynamicProxyClientAsyncPolling {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DynamicProxyClientAsyncPolling");

      System.out.println("Creating proxy instance for service StockQuoteAxisService");
      StockQuoteAxisService stub = new StockQuoteAxisService();
      StockQuoteAxis service = stub.getSoapServerStockQuoteAxisWmq();

      System.out
          .println("Invoking getQuoteAsync Request Reply operation asynchronously by
polling...");
      Response<Float> response = service.getQuoteAsync("49");

      /** Sleep main thread until response arrives **/
      System.out.println("Waiting for response to arrive...");
      while (!response.isDone()) {
        Thread.sleep(100);
      }
      System.out.println(" > Response received");

      /** Retrieve the result **/
      try {
        Float result = response.get();
        System.out.println(" > getQuoteAsync call has returned result of " + result);
      }
      catch (CancellationException ce) {
        // processing was cancelled via response.cancel()
      }

      System.out.println("End of sample");
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
```

```
  }
}
```

Şekil 186. DynamicProxyClientAsyncCallback.java

```
package com.ibm.mq.axis2.samples;

import java.util.concurrent.Future;

import javax.xml.ws.AsyncHandler;
import javax.xml.ws.Response;

import com.ibm.mq.axis2.samples.proxy.StockQuoteAxis;
import com.ibm.mq.axis2.samples.proxy.StockQuoteAxisService;

public class DynamicProxyClientAsyncCallback implements AsyncHandler<Float> {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DynamicProxyClientAsyncCallback");

      System.out.println("Creating proxy instance for service StockQuoteAxisService");
      StockQuoteAxisService stub = new StockQuoteAxisService();
      StockQuoteAxis service = stub.getSoapServerStockQuoteAxisWmq();

      DynamicProxyClientAsyncCallback handler = new DynamicProxyClientAsyncCallback();

      System.out
          .println("Invoking getQuoteAsync Request Reply operation asynchronously using a
callback...");
      Future<?> monitor = service.getQuoteAsync("50", handler);
      System.out.println(" > Invoke call has returned");

      /** Sleep main thread until handler has been notified **/
      System.out.println("Waiting for handler to be called...");
      while (!monitor.isDone()) {
        Thread.sleep(100);
      }

      System.out.println("End of sample");
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }

  public void handleResponse(Response<Float> response) {
    try {
      Float result = response.get();
      System.out.println(" > Async Handler has received a result of " + result);
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println("Exception in handleResponce");
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());
```

```
        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }

}
```

Şekil 187. DispatchClientSync.java

```
package com.ibm.mq.axis2.samples;

import javax.xml.namespace.QName;
import javax.xml.soap.MessageFactory;
import javax.xml.soap.SOAPBody;
import javax.xml.soap.SOAPConstants;
import javax.xml.soap.SOAPElement;
import javax.xml.soap.SOAPEnvelope;
import javax.xml.soap.SOAPHeader;
import javax.xml.soap.SOAPMessage;
import javax.xml.soap.SOAPPart;
import javax.xml.ws.Dispatch;
import javax.xml.ws.Service;
import javax.xml.ws.soap.SOAPBinding;

public class DispatchClientSync {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DispatchClientSync");

      String endpointUrl = "jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM&"
          + "connectionFactory=connectQueueManager(WMQSOAP.DEMO.QM)"
          +
"&initialContextFactory=com.ibm.mq.jms.Nojndi&targetService=soap.server.StockQuoteAxis.java";

      QName serviceName = new QName("soap.server.StockQuoteAxis_Wmq", "StockQuoteAxisService");
      QName portName = new QName("soap.server.StockQuoteAxis_Wmq",
"soap.server.StockQuoteAxis_Wmq");

      Service service = Service.create(serviceName);
      service.addPort(portName, SOAPBinding.SOAP11HTTP_BINDING, endpointUrl);

      /** Create a Dispatch instance from a service **/
      System.out.println("Creating dispatch instance for service StockQuoteAxisService");
      Dispatch<SOAPMessage> dispatch = service.createDispatch(portName, SOAPMessage.class,
          Service.Mode.MESSAGE);
      System.out.println(" > Dispatch instance created.");

      /************************************
       * Create OneWay SOAPMessage request.
       ************************************/
      MessageFactory mf = MessageFactory.newInstance(SOAPConstants.SOAP_1_1_PROTOCOL);

      System.out.println("\nCreating a OneWay SOAP Message");
      SOAPMessage request = mf.createMessage();

      /** Obtain the SOAPEnvelope and header and body elements **/
      SOAPPart part = request.getSOAPPart();
      SOAPEnvelope env = part.getEnvelope();
      SOAPHeader header = env.getHeader();
      SOAPBody body = env.getBody();

      /** Construct the message payload **/
      SOAPElement operation = body.addChildElement("getQuoteOneWay", "ns1",
          "soap.server.StockQuoteAxis_Wmq");
      SOAPElement value = operation.addChildElement("in0");
      value.addAttribute(new QName("https://www.w3.org/2001/XMLSchema-instance", "type"),
"string");
      value.addTextNode("XXX");
      request.saveChanges();
      System.out.println(" > SOAP Message created.");
```

```
      /** Invoke the service endpoint **/
      System.out.println("Invoking getQuoteOneWay OneWay operation synchronously...");
      dispatch.invokeOneWay(request);
      System.out.println(" > getQuoteOneWay call has returned");

      /*****************************************
       * Create Request Reply SOAPMessage request.
       *****************************************/
      mf = MessageFactory.newInstance(SOAPConstants.SOAP_1_1_PROTOCOL);

      System.out.println("\nCreating a Request Reply SOAP Message");
      request = mf.createMessage();

      /** Obtain the SOAPEnvelope and header and body elements **/
      part = request.getSOAPPart();
      env = part.getEnvelope();
      header = env.getHeader();
      body = env.getBody();

      /** Construct the message payload **/
      operation = body.addChildElement("getQuote", "ns1", "soap.server.StockQuoteAxis_Wmq");
      value = operation.addChildElement("in0");
      value.addAttribute(new QName("https://www.w3.org/2001/XMLSchema-instance", "type"),
"string");
      value.addTextNode("XXX");
      request.saveChanges();
      System.out.println(" > SOAP Message created.");

      /** Invoke the service endpoint **/
      System.out.println("Invoking getQuote Request Reply operation synchronously...");
      SOAPMessage ans = dispatch.invoke(request);
      System.out.println(" > getQuote call has returned");

      /** Retrieve the result **/
      part = ans.getSOAPPart();
      env = part.getEnvelope();
      body = env.getBody();

      /** Define name of the SOAP folders we are interested in **/
      QName responseName = new QName("soap.server.StockQuoteAxis_Wmq", "getQuoteResponse");
      QName resultName = new QName("getQuoteReturn");

      /** Retrieve result from SOAP envelope **/
      System.out.println("Parsing SOAP response...");
      SOAPElement bodyElement = (SOAPElement) body.getChildElements(responseName).next();
      SOAPElement responseElement = (SOAPElement)
bodyElement.getChildElements(resultName).next();
      String message = responseElement.getValue();
      System.out.println(" > Response contains result of " + message);

      System.out.println("End of sample");
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }
}
```

Şekil 188. DispatchClientAsyncPolling.java

```
package com.ibm.mq.axis2.samples;
```

```
import javax.xml.namespace.QName;
import javax.xml.soap.MessageFactory;
import javax.xml.soap.SOAPBody;
import javax.xml.soap.SOAPConstants;
import javax.xml.soap.SOAPElement;
import javax.xml.soap.SOAPEnvelope;
import javax.xml.soap.SOAPHeader;
import javax.xml.soap.SOAPMessage;
import javax.xml.soap.SOAPPart;
import javax.xml.ws.Dispatch;
import javax.xml.ws.Response;
import javax.xml.ws.Service;
import javax.xml.ws.soap.SOAPBinding;

public class DispatchClientAsyncPolling {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DispatchClientAsyncPolling");

      String endpointUrl = "jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM&"
          + "connectionFactory=connectQueueManager(WMQSOAP.DEMO.QM)"
          +
"&initialContextFactory=com.ibm.mq.jms.Nojndi&targetService=soap.server.StockQuoteAxis.java";

      QName serviceName = new QName("soap.server.StockQuoteAxis_Wmq", "StockQuoteAxisService");
      QName portName = new QName("soap.server.StockQuoteAxis_Wmq",
"soap.server.StockQuoteAxis_Wmq");

      Service service = Service.create(serviceName);
      service.addPort(portName, SOAPBinding.SOAP11HTTP_BINDING, endpointUrl);

      /** Create a Dispatch instance from a service.* */
      System.out.println("Creating dispatch instance for service StockQuoteAxisService");
      Dispatch<SOAPMessage> dispatch = service.createDispatch(portName, SOAPMessage.class,
          Service.Mode.MESSAGE);
      System.out.println(" > Dispatch instance created.");

      /** Create SOAPMessage request. * */
      MessageFactory mf = MessageFactory.newInstance(SOAPConstants.SOAP_1_1_PROTOCOL);

      System.out.println("Creating a Request Reply SOAP Message");
      SOAPMessage request = mf.createMessage();

      /** Obtain the SOAPEnvelope and header and body elements **/
      SOAPPart part = request.getSOAPPart();
      SOAPEnvelope env = part.getEnvelope();
      SOAPHeader header = env.getHeader();
      SOAPBody body = env.getBody();

      /** Construct the message payload **/
      SOAPElement operation = body.addChildElement("getQuote", "ns1",
          "soap.server.StockQuoteAxis_Wmq");
      SOAPElement value = operation.addChildElement("in0");
      value.addAttribute(new QName("https://www.w3.org/2001/XMLSchema-instance", "type"),
"string");
      value.addTextNode("XXX");
      request.saveChanges();
      System.out.println(" > SOAP Message created.");

      /** Invoke the service endpoint **/
      System.out.println("Invoking getQuote Request Reply operation asynchronously by
polling...");
      Response<SOAPMessage> response = dispatch.invokeAsync(request);
      System.out.println(" > getQuote call has returned");

      /** Sleep main thread until response arrives **/
      System.out.println("Waiting for response to arrive...");
      while (!response.isDone()) {
        Thread.sleep(100);
      }
      System.out.println(" > Response received");

      /** retrieve the result **/
      SOAPMessage ans = response.get();
      part = ans.getSOAPPart();
      env = part.getEnvelope();
      body = env.getBody();

      /** Define name of the SOAP folders we are interested in **/
      QName responseName = new QName("soap.server.StockQuoteAxis_Wmq", "getQuoteResponse");
      QName resultName = new QName("getQuoteReturn");
```

```
      /** Retrieve result from SOAP envelope **/
      SOAPElement bodyElement = (SOAPElement) body.getChildElements(responseName).next();
      SOAPElement responseElement = (SOAPElement)
bodyElement.getChildElements(resultName).next();
      String message = responseElement.getValue();
      System.out.println(" > Response contains result of " + message);

      System.out.println("End of sample");

    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }
}
```

Şekil 189. DispatchClientAsyncCallback.java

```
package com.ibm.mq.axis2.samples;

import java.util.concurrent.Future;

import javax.xml.namespace.QName;
import javax.xml.soap.MessageFactory;
import javax.xml.soap.SOAPBody;
import javax.xml.soap.SOAPConstants;
import javax.xml.soap.SOAPElement;
import javax.xml.soap.SOAPEnvelope;
import javax.xml.soap.SOAPHeader;
import javax.xml.soap.SOAPMessage;
import javax.xml.soap.SOAPPart;
import javax.xml.ws.AsyncHandler;
import javax.xml.ws.Dispatch;
import javax.xml.ws.Response;
import javax.xml.ws.Service;
import javax.xml.ws.soap.SOAPBinding;

public class DispatchClientAsyncCallback implements AsyncHandler<SOAPMessage> {

  public static void main(String[] args) {
    try {
      System.out.println("Starting sample DispatchClientAsyncCallback");

      String endpointUrl = "jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM&"
          + "connectionFactory=connectQueueManager(WMQSOAP.DEMO.QM)"
          +
"&initialContextFactory=com.ibm.mq.jms.Nojndi&targetService=soap.server.StockQuoteAxis.java";

      QName serviceName = new QName("soap.server.StockQuoteAxis_Wmq", "StockQuoteAxisService");
      QName portName = new QName("soap.server.StockQuoteAxis_Wmq",
"soap.server.StockQuoteAxis_Wmq");

      Service service = Service.create(serviceName);
      service.addPort(portName, SOAPBinding.SOAP11HTTP_BINDING, endpointUrl);

      /** Create a Dispatch instance from a service.* */
      System.out.println("Creating dispatch instance for service StockQuoteAxisService");
      Dispatch<SOAPMessage> dispatch = service.createDispatch(portName, SOAPMessage.class,
          Service.Mode.MESSAGE);
      System.out.println(" > Dispatch instance created.");
```

```
      /** Create SOAPMessage request. * */
      MessageFactory mf = MessageFactory.newInstance(SOAPConstants.SOAP_1_1_PROTOCOL);

      System.out.println("Creating a Request Reply SOAP Message");
      SOAPMessage request = mf.createMessage();

      /** Obtain the SOAPEnvelope and header and body elements **/
      SOAPPart part = request.getSOAPPart();
      SOAPEnvelope env = part.getEnvelope();
      SOAPHeader header = env.getHeader();
      SOAPBody body = env.getBody();

      /** Construct the message payload. **/
      SOAPElement operation = body.addChildElement("getQuote", "ns1",
          "soap.server.StockQuoteAxis_Wmq");
      SOAPElement value = operation.addChildElement("in0");
      value.addAttribute(new QName("https://www.w3.org/2001/XMLSchema-instance", "type"),
"string");
      value.addTextNode("XXX");
      request.saveChanges();
      System.out.println(" > SOAP Message created.");

      /** Invoke the service endpoint. **/
      DispatchClientAsyncCallback handler = new DispatchClientAsyncCallback();

      System.out
          .println("Invoking getQuote Request Reply operation asynchronously using a
callback...");
      Future<?> monitor = dispatch.invokeAsync(request, handler);
      System.out.println(" > getQuote call has returned");

      /** Sleep main thread until handler has been notified **/
      System.out.println("Waiting for handler to be called...");
      while (!monitor.isDone()) {
        Thread.sleep(100);
      }

      System.out.println("End of sample");
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }

  public void handleResponse(Response<SOAPMessage> response) {
    try {
      // retrieve the result
      SOAPMessage ans = response.get();
      SOAPPart part = ans.getSOAPPart();
      SOAPEnvelope env = part.getEnvelope();
      SOAPBody body = env.getBody();

      /** Define name of the SOAP folders we are interested in **/
      QName responseName = new QName("soap.server.StockQuoteAxis_Wmq", "getQuoteResponse");
      QName resultName = new QName("getQuoteReturn");

      /** Retrieve result from SOAP envelope **/
      SOAPElement bodyElement = (SOAPElement) body.getChildElements(responseName).next();
      SOAPElement responseElement = (SOAPElement)
bodyElement.getChildElements(resultName).next();
      String result = responseElement.getValue();

      System.out.println(" > Async Handler has received a result of " + result);
    }
    catch (Exception fault) {
      // Identify the cause of the Axis Fault
      System.err.println("Exception in handleResponce");
```

```
      System.err.println(fault.toString());
      Throwable e = fault.getCause();
      for (int i = 1; e != null; i++) {
        // The toString method on an MQAxisException will cause the message, explanation and
user
        // action.
        System.err.println("Exception(" + i + "): " + e.toString());

        if (e.getCause() != null) {
          e = e.getCause();
        }
        else {
          break;
        }
      } // end of for loop
    } // end of catch block
  }

}
```

İlgili görevler

Developing a JAX-RPC client for WebSphere transport for SOAP using Eclipse
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti istemcisi geliştirin.

Developing a .NET 1 or 2 client for WebSphere transport for SOAP using Microsoft Visual Studio 2008
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir .NET 1 ya da 2 Web hizmeti istemcisi geliştirin.

Developing a .NET 1 or 2 client for WebSphere transport for SOAP using Microsoft Visual Studio 2008

SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir .NET 1 ya da 2 Web hizmeti istemcisi geliştirin.

Başlamadan önce

.NET 1 ya da 2 istemcisinin geliştirilmesini çeşitli yollarla başlatabilirsiniz:

1. Bir Web hizmetinden istemci sınırlı kod öbekleri oluşturmak ve bunları Visual Studio 'ya aktarmak için **amqwdeployWMQService** ' ı kullanın.
2. Bir Web hizmetinin Java uygulamasından WSDL oluşturmak için **java2wsdl** 'u kullanın ve sonra istemci sınırlı kod öbekleri oluşturmak için .NET ile birlikte gönderilen `wsdl.exe`' i kullanın.
3. **amqswsdl**kullanan hizmetin bir .NET `.asmx` uygulamasından WSDL ' yi oluşturun ve sonra `wsdl.exe`komutunu kullanın.
4. HTTP için hizmeti geliştirdiyseniz ve konuşlandırdıysanız, **Web başvurusu ekle ...** sihirbazı Visual Studio 'da, istemciyi HTTP hizmetine erişecek şekilde yapılandırır. URL ' yi değiştirin, SOAP için WebSphere MQ iletimi için konuşlandırılan hizmete bakın.

Görev, “Developing a .NET 1 or 2 service for WebSphere MQ transport for SOAP using Microsoft Visual Studio 2008” sayfa 926içinde geliştirilen hizmeti kullanır.

Bu görev hakkında

SOAP için .NET 1 ya da 2 Client for HTTP ve WebSphere MQ iletimi yaratmak için aşağıdaki adımları izleyin.

Yordam

1. İstemci konsolu uygulamasını yaratın ve StockQuote HTTP Web hizmetini çağırmak için bu uygulamayı değiştirin.
   a) **Solution Explorer** > Add ... > New Project öğelerini kullanarak **Solution 'StockQuoteDotNet'** öğesini farenin sağ düğmesiyle tıklatın. **C#** Project Type, **.NET Framework 2.0**ve **Konsol Uygulaması**öğelerini seçin. Projeyi adı `StockQuoteClientDotNet` > **Tamam**

b) **Solution Explorer** > Add ... > New Project öğelerini kullanarak **Solution 'StockQuoteDotNet'** öğesini farenin sağ düğmesiyle tıklatın. **C#** Project Type, **.NET Framework 2.0**ve **Konsol Uygulaması**öğelerini seçin. Projeyi adı `StockQuoteClientDotNet` > **Tamam**

c) **StockQuoteClientDotNet** > **Set as Startup projesi**seçeneğini sağ tıklatın.

d) **StockQuoteClientDotNet** > **Add Web Reference ...** > Browse to Web services in this solution > Select **StockQuote** > **Add Reference**öğelerini sağ tıklatın. Yerel anasisteme ve yeni bir yapılandırma dosyasına (`app.config`) bir Web başvurusu eklediğinizi fark edin.

e) In the Solution Explorer, change the name of the console application from `Program.cs` to `StockQuoteClientDotNet.cs` > Click **Tamam** to changing all the usages of `Program.cs` to `StockQuoteClientDotNet.cs`.

f) `StockQuoteClientDotNet.cs` içeriğini Şekil 190 sayfa 957kodundaki kodla değiştirin.

```
using System;
using StockQuoteClientDotNet.localhost;
namespace StockQuoteClientDotNet {
  class StockQuoteClientDotNet {
    static void Main(string[] args) {
      try {
        StockQuote stockobj = new StockQuote();
        Console.WriteLine("http reply is: "
          + stockobj.getNonInlineQuote("http request");
      }
      catch (System.Exception e) {
        Console.WriteLine("Exception thrown: " + e.ToString());
      }
      Console.ReadLine();
    }
  }
}
```

Şekil 190. HTTP StockQuoteClientDotNet programı

g) `StockQuote.asmx` hizmetine karşı test etmek için StockQuoteClientDotNet 'i başlatın:

i) **F5**tuşuna basın, işlem çubuğundaki yeşil oku ya da **Hata Ayıkla** > **Hata Ayıklamayı Başlat (F5)**düğmesini tıklatın.

StockQuoteDOtNet projesi aynı çözümde olduğunda, otomatik olarak başlatılır. Tersi durumda, önce hizmeti başlatmanız gerekir.

Sonuçlarla birlikte komut penceresi çalışma alanının arkasında açılır. `Console.ReadLine();` deyimi, **Enter**tuşuna basana kadar kapanmasını önler.

İpucu: `StockQuote.asmx` ' un `StockQuoteDotNet` projesindeki Başlangıç sayfası olduğundan emin olun.

2. Modify StockQuoteClientDotNet to call the `StockQuote.asmx` service using WebSphere MQ transport for SOAP.

a) Kalın olarak gösterilen satırları istemciye ekleyin.

```
using System;
using StockQuoteClientDotNet.localhost;
namespace StockQuoteClientDotNet {
  class StockQuoteClientDotNet {
    static void Main(string[] args) {
      try {
        IBM.WMQSOAP.Register.Extension();
        StockQuote stockobj = new StockQuote();
        Console.WriteLine("http reply is: "
          + stockobj.getNonInlineQuote("http request");
        stockobj.Url = "jms:/queue?"
          + "initialContextFactory=com.ibm.mq.jms.Nojndi"
          + "&connectionFactory=()&destination=REQUESTDOTNET@QM1"
          + "&targetService=StockQuote.asmx";
        Console.WriteLine("jms reply is: "
          + stockobj.getNonInlineQuote("jms request"));
      }
      catch (System.Exception e) {
        Console.WriteLine("Exception thrown: " + e.ToString());
      }
      Console.ReadLine();
    }
  }
}
```

Şekil 191. Değiştirilen StockQuoteClientDotNet programı

Diğer bir seçenek olarak, varsayılan URL ' yi değiştirin. Open **StockQuoteClientDotNet** > **Özellikler** > **Settings.settings** and change the value of the `StockQuoteClientDotNet_localhost_StockQuote` property to the WebSphere MQ transport for SOAP URL.

b) `amqsoap.dll`öğesine bir başvuru ekleyin

i) In the **StockQuoteClientDotNet** project in the **Çözüm Gezgini**, right-click **Başvurular** > **Başvuru Ekle ...** > Click the **Göz At** tab > browse to *MQ_INSTALLATION_PATH*\`bin` > Select **amqsoap.dll** > **Tamam**. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir.

3. SOAP için WebSphere MQ iletimi kullanan istemciyi `StockQuote.asmx` hizmetiyle test edin.

a) StockQuoteDotNet proje dizininde bir komut penceresi açın: `.\StockQuoteDotNet\StockQuoteDotNet` > `.bin\StockQuoteDotNet.dll` ' un var olduğunu doğrulayın. Bu durumda değilse, çözümü yeniden oluşturun.

b) **amqwRegisterdotNet**komutunu yazın.

Kuruluş başına yalnızca **amqwRegisterdotNet** ' yi çalıştırmanız gerekir.

c) **amqwdeployWMQServer** ' u `genAsmxWMQBits`ile çalıştırdıysanız, .NET SOAP Listener 'ı çalıştırın: `generated\server\startWMQNListener`

d) Diğer bir seçenek olarak, dinleyiciyi doğrudan çalıştırın

```
amqwSOAPNETListener -u "jms:/queue?
destination=REQUESTDOTNET@QM1
&connectionFactory=()&initialContextFactory=com.ibm.mq.jms.Nojndi
&targetService=StockQuote.asmx&replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE"
-w C:\IBM\ID\StockQuoteDotNet\StockQuoteDotNet -n 10
```

4. Visual Studio 2008 'de StockQuoteClientDotNet 'i çalıştırmak için **F5** tuşuna basın.

.NET Framework 1 ve .NET Framework 2 Web hizmeti istemcileri

SOAP kullanımına ilişkin WebSphere MQ iletimi ile sağlanan örnek .NET istemcileri, örnek Axis ve .NET hizmetlerini çağırmak için kullanılan sınırlı kod öbeklerini kullanır.

.NET Framework 1 ve .NET Framework 2 istemcileri için, WebSphere MQ , .NET istemcilerini kullanarak web hizmetlerine erişim sağlar. **amqwdeployWMQService** komutu, bir Web hizmeti için .NET Framework 1 ya da .NET Framework 2 istemci sınırlı kod öbekleri üreten bir seçeneği (`genProxiestoDotNet`)

içerir. .NET **wsdl** aracı tarafından oluşturulan istemci sınırlı kod öbeklerini ya da Microsoft Visual Studio 2005 ya da 2008 'i de kullanabilirsiniz.

Örnek .NET Framework 1 ve .NET Web hizmeti istemcileri, *MQ_INSTALLATION_PATH*\tools\soap\samples\dotnetiçine kurulur. *MQ_INSTALLATION_PATH* WebSphere MQ ' un kurulu olduğu dizindir.

SQVB2Axis.vb
SQVB2Axis.vb, Şekil 192 sayfa 959, **StockQuoteAxisService** hizmetini çağırmak için Visual Basic istemcisidir.

SQVB2DotNet.vb
QVB2DotNet.vb, Şekil 193 sayfa 959, **StockQuoteDotNet** hizmetini çağırmak için Visual Basic istemcisidir.

SQCS2Axis.cs
SQCS2Axis.cs, Şekil 194 sayfa 959, is the C# client to call the **StockQuoteAxisService** service. Komut satırında bir URL sağlayarak hizmetin URL adresini geçersiz kılabilirsiniz.

SQCS2DotNet.cs
SQCS2DotNet.cs, Şekil 195 sayfa 960, is the C# client to call the **StockQuoteDotNet** service. Komut satırında bir URL sağlayarak hizmetin URL adresini geçersiz kılabilirsiniz.

```
Module SQVB2Axis
   Function Main(ByVal CmdArgs() As String) As Integer
      IBM.WMQSOAP.Register.Extension()
      Dim obj As New StockQuoteAxisService()
      Dim res As Single = obj.getQuote("fromcs")
      System.Console.WriteLine("SQVB2Axis: reply is: '{0}'", res)
   End Function
End Module
```

Şekil 192. SQVB2Axis

```
Module SQVB2DotNet
   Function Main(ByVal CmdArgs() As String) As Integer
      IBM.WMQSOAP.Register.Extension()
      Dim obj as new StockQuoteDotNet()
      Dim res as Single = obj.getQuote("fromcs")
      System.Console.WriteLine("SQVB2DotNet: reply is: '{0}'", res)
   End Function
End Module
```

Şekil 193. SQVB2DotNet

```
using System;
class SQCS2Axis {
   [STAThread]
   static void Main(string[] args) {
      try {
         IBM.WMQSOAP.Register.Extension();
         StockQuoteAxisService stockobj = new StockQuoteAxisService();
         if (args.GetLength(0) >= 1)
            stockobj.Url = args[0];
         System.Single res = stockobj.getQuote("XXX");
         Console.WriteLine("SQCS2Axis RPC reply is: " + res);
      }
      catch (System.Exception e) {
         Console.WriteLine("\n>>> EXCEPTION WHILE RUNNING SQCS2Axis DEMO <<<\n"
            + e.ToString());
      }
   }
}
```

Şekil 194. SQCS2Axis

```
using System;
class SQCS2DotNet {
   [STAThread]
   static void Main(string[] args) {
      try {
         IBM.WMQSOAP.Register.Extension();
         StockQuoteDotNet stockobj = new StockQuoteDotNet();
         if (args.GetLength(0) >= 1)
            stockobj.Url = args[0];
         System.Single res = stockobj.getQuote("XXX");
         Console.WriteLine("RPC reply is: " + res);
         if (args.GetLength(0) == 0) {
            res = stockobj.getQuoteDOC("XXX");
            Console.WriteLine("DOC reply is: " + res);
         }
      }
      catch (System.Exception e) {
         Console.WriteLine("\n>>> EXCEPTION WHILE RUNNING SQCS2DotNet DEMO <<<\n"
            + e.ToString());
      }
   }
}
```

Şekil 195. SQCS2DotNet

İlgili görevler

Developing a JAX-RPC client for WebSphere transport for SOAP using Eclipse
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Eksen 1.4 Web hizmeti istemcisi geliştirin.

Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse
SOAP için WebSphere MQ iletimi kullanarak çalıştırmak için bir Axis2 Web hizmeti istemcisi geliştirin. SOAP için WebSphere MQ iletimiyle sağlanan örnek Axis2 istemcileri listelenir ve yetkili sunucular oluşturmak için kullanılan **wsimport** komutu kullanılır.

SOAP için WebSphere MQ iletimi kullanılarak Web hizmetlerinin konuşlandırılması

Bir Web hizmetini farklı sunucu ortamlarından birine konuşlandırın ve SOAP için WebSphere MQ iletimi olanağını kullanarak bu hizmeti kullanın.

Başlamadan önce

Bir Web hizmeti geliştirin ve hedef ortamda HTTP üzerinden SOAP kullanarak test edin.

Bu görev hakkında

Bir dizi farklı SOAP çalıştırma zamanı ortamında SOAP için WebSphere MQ iletimi ile çalıştırmak üzere bir web hizmeti konuşlandırabilirsiniz. Yalnızca WebSphere MQile kurulan yazılımı kullanarak 1.4 eksenine bir hizmet dağıtabilirsiniz. Diğer çalıştırma zamanı ortamları için ek yazılım kurmalısınız.

Konuşlandırma yönergelerinin bulunduğu sunuculara SOAP için WebSphere MQ aktarımı çalıştırılmayla kısıtlanmadınız. Listelenen ortamlardan birine hizmet konuşlandırmak için yönergeleri kullanın.

Not: Some integrated environments offer SOAP over JMS using the W3C recommended JMS SOAP binding, as well as the WebSphere MQ transport for SOAP binding. Releases of WebSphere MQ, up to and including 7.0.1.2, support only the WebSphere MQ transport for SOAP binding. 7.0.1.3 ' tan başlayarak, JMS üzerinden SOAP için W3C aday önerisine uygun bir URI kullanarak Axis2 istemcilerini konuşlandırabilirsiniz. See the tutorial, WebSphere Application Server V7 ve Rational Application Developer V7.5ile SOAP/JMS JAX-WS Web hizmetleri uygulaması geliştirin.

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService

Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

Başlamadan önce

1. SOAP için WebSphere MQ iletimi kurmaya ilişkin yönergeleri izleyin.
2. **runivt** komutunu kullanarak kuruluşu ve ortamınızı doğrulayın.
3. Bir hizmeti yeniden konuşlandırmak için:
   a. `./generated` alt dizinini ve tüm alt dizinlerini silin.
   b. Hedef kuyruktan gelen istekleri kaldırın ve silin.
   c. “2” sayfa 961. adımdaki yönergelerle devam edin.

Bu görev hakkında

Bu yönergeler, bir Axis 1.4 hizmetini ilk kez devreye almandır. Bir Axis 1.4 hizmetini yeniden başlatmak için, Axis 1.4 SOAP dinleyicisini yeniden çalıştırın: adım “11” sayfa 962.

SOAP için WebSphere MQ iletimi için yeni bir Axis 1.4 hizmetini konuşlandırmak için aşağıdaki yönergeleri kullanın:

Yordam

1. Konuşlandırma dosyalarını tutmak için *deployDir* dizini oluşturun.

   Devreye alma yardımcı programı, her hizmetin ayrı bir dizinden dağıtılmasını gerektirir.
2. Open a command window on Windows, or a command shell using X Window System on UNIX and Linux systems, in *deployDir* to run **amqwdeployWMQService**.
3. Sınıf yolunu ayarlamak için **amqwsetcp** komutunu çalıştırın.

   JRE ve JDK, sınıf yolunda (classpath), sürüm 5.0 ya da sonraki düzeyler ve aynı sürüm düzeyinde olmalıdır.
4. Sınıf kaynağını (*className*`.java`) *deployDir*içine kopyalayın.
5. Aynı paketteki tüm Java kaynak dosyalarını *className* ile *deployDir*/*packageName*içine kopyalayın; burada *packageName* , paket adına karşılık gelen bir dizin ağacidir.
6. Çalıştır **javac** *packageName*.*className*.

   Diğer sınıfları bulmak için, yürürlükteki dizine ("`.`") ya da **javac** dizinine ilişkin *packageName* dizinine bir yol eklemeniz gerekebilir.
7. Hizmet için Eksen WSDL ' yi yaratın:

   ```
   amqwdeployWMQService -f packageName.className.java -c genAxisWsdl
   -v -u  "jms:/queue?destination=queueName
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &connectionFactory=(connectQueueManager(QmgrName)binding(auto))"
   ```
8. Hizmet için WebSphere MQ kaynakları yaratın:

   ```
   amqwdeployWMQService -f packageName.className.java -c genAxisWMQBits
   -v -u  "jms:/queue?destination=queueName
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &connectionFactory=(connectQueueManager(QmgrName)binding(auto))"
   ```

   İpucu:

   Yeni bir kuyruk yöneticisi ve gereksinim duyduğu kaynakları kurmak istiyorsanız, geliştirme ve test etme işlemi için **setupWMQSOAP**komutunu çalıştırın.

   If you want set up the new queue manager as the default, take a copy of **setupWMQSOAP** from the *WMQ install directory*`\tools\soap\samples` directory, and add the `-q` parameter to the line

   ```
   call :try -q crtmqm %QMGR%
   ```
9. Axis dinleyicisini yaratın ve hizmeti konuşlandırın:

```
amqwdeployWMQService -f packageName.className.java -c AxisDeploy
-v -u  "jms:/queue?destination=queueName
&initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=(connectQueueManager(QmgrName)binding(auto))"
```

10. Hizmet için WSDL ' yi oluşturmanız, istemci sınırlı kod öbekleri ya da istemci yetkili sunucuları oluşturmanız gerekiyorsa, **amqwdeployWMQService** komutunu aşağıdaki değiştirgelerden biriyle çalıştırın:

    - genAsmxWsdl
    - genAxisWsdl
    - genProxiesToDotNet
    - genProxiestoEkseni

    Not: Yetkili sunucuları oluşturmadan önce WSDL oluşturmalısınız. The AllAxis option fails if the CLAS is not set up to find all the classes that are imported to compile *className*.java. If there are multiple Java files in the package containing *className*.java, you must compile them first using **javac**. **amqwdeployWMQService** -f *packageName*.*className*.java -c CompileJava compiles only *className*.java.

11. Oluşturulan Eksen dinleyicisini başlatın.

```
.\generated\server\startWMQJListener.cmd
```

İlgili görevler

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması
SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması
SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması
WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Configuring WebSphere Application Server to use W3C SOAP over JMS
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması
WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması

SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

Başlamadan önce

1. SOAP için WebSphere MQ iletimi kurmaya ilişkin yönergeleri izleyin.
2. **runivt** komutunu kullanarak kuruluşu ve ortamınızı doğrulayın.

3. The path to the .NET framework files `wsdl.exe` and `csc.exe` must be set. PATH değişkeniyle tanıtılan `wsdl.exe` ve `csc.exe` kopyaları, .NET çerçevesinin aynı düzeyinde olmalıdır. Sisteminizde birden çok .NET frameworks kuruluysa ya da Visual Studio kullanıyorsanız, YOL değişkenini dikkatli bir şekilde denetleyin.
4. Bir hizmeti yeniden konuşlandırmak için:
   a. `./generated` alt dizinini ve tüm alt dizinlerini sil
   b. Hedef kuyruktan gelen istekleri kaldırın ve silin.
   c. “2” sayfa 963. adımdaki yönergelerle devam edin.

Bu görev hakkında

Bu yönergeler, .NET hizmetini ilk kez konuşlandırmanız içindir. Bir .NET hizmetini yeniden başlatmak için, .NET SOAP dinleyicisini yeniden çalıştırın; adım “9” sayfa 964.

SOAP için yeni bir .NET Framework 1 ya da .NET Framework 2 hizmetini WebSphere MQ iletimi için konuşlandırmak için aşağıdaki yönergeleri kullanın:

Yordam

1. Konuşlandırma dosyalarını tutmak için *deployDir* dizini oluşturun.
   Devreye alma yardımcı programı, her hizmetin ayrı bir dizinden dağıtılmasını gerektirir.
2. **amqwdeployWMQService**komutunu çalıştırmak için *deployDir* içinde bir komut penceresi açın.

```
C:\IBM\ID\QuoteClient>
```

3. Sınıf yolunu ayarlamak için **amqwsetcp** komutunu çalıştırın.
   Yalnızca Eksen istemcileri için bir sınıf yolu (classpath) gereklidir.
4. .NET hizmetini (*className*`.asmx`) *deployDir*içine kopyalayın.
5. Hizmet somutlamasını bir kitaplığa oluşturun (`.dll`).

   İç hizmet somutlaması *className*`.asmx`içinde yer alıyor. Kod arkasındaki hizmet somutlaması *className*`.asmx.cs`olabilir.

   Şekil 196 sayfa 963 , bir .NET Framework V2 hizmetini kitaplık olarak oluşturmak için bir komut örneği gösterir.

```
c:\WINDOWS\Microsoft.NET\Framework\v3.5\Csc.exe /noconfig /nowarn:1701,1702
/errorreport:prompt /warn:4 /define:TRACE
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.configuration.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Data.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Drawing.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Web.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Web.Services.dll
/reference:c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Xml.dll
/debug:pdbonly /filealign:512 /optimize+
/out:obj\Quote.dll /target:library Properties\AssemblyInfo.cs Quote.asmx.cs
```

Şekil 196. .NET Framework V2 hizmeti için oluşturma komutu

6. *className*`.dll` dosyasını *deployDir*`\bin`içine kopyalayın.
7. WebSphere MQ kaynaklarını ayarlayın ve hizmet için gereken dinleyiciyi yaratın:

```
amqwdeployWMQService -f className.asmx -c genAsmxWMQBits
-v -u  "jms:/queue?destination=queueName
&initialContextFactory=com.ibm.mq.jms.Nojndi
&connectionFactory=(connectQueueManager(QmgrName)binding(auto))
&targetService=className.asmx"
```

8. Hizmet için WSDL ' yi oluşturmanız, istemci sınırlı kod öbekleri ya da istemci yetkili sunucuları oluşturmanız gerekiyorsa, **amqwdeployWMQService** komutunu aşağıdaki değiştirgelerden biriyle çalıştırın:
   - `genAsmxWsdl`
   - `genAxisWsdl`
   - `genProxiesToDotNet`
   - `genProxiestoEkseni`

   Not: Yetkili sunucuları oluşturmadan önce WSDL oluşturmalısınız.
9. Üretilen .NET dinleyicisini başlatın.

```
.\generated\server\startWMQNListener.cmd
```

İlgili görevler

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService
Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması
SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması
WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Configuring WebSphere Application Server to use W3C SOAP over JMS
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması
WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması

SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

Başlamadan önce

HTTP için geliştirdiğiniz gibi, WebSphere MQiçin bir istemci ya da hizmet için geliştirmek üzere aynı araçları kullanın. CICS , **Java2wsdl** ve **wsdl2Java**' ye karşılık gelen araçlara sahiptir:

- **DFHWS2LS** , bir Web hizmeti tanımlamasını başlangıç noktası olarak alır. Yüksek düzeyli dil veri yapıları oluşturmak için bu iletilerde kullanılan veri tiplerini ve bu iletilerde kullanılan veri tiplerini kullanır. Farklı dillerde yazılmış uygulama programlarındaki yapılarda da kullanabilirsiniz.
- **DFHLS2WS** , başlangıç noktası olarak üst düzey bir dil veri yapısını alır. İletilerin açıklamalarını içeren bir Web hizmetleri tanımı oluşturmak için yapıyı kullanır. Ayrıca, dil veri yapısındaki iletiler için şemalar yaratır.

Web hizmeti yaratmak için, CICS ürün belgelerinde Creating a Web service (Web hizmeti yaratılması) başlıklı yönergeleri izleyin.

Bu görev hakkında

Follow the instructions, CICS ' in WebSphere MQ iletisini kullanmak için yapılandırılması in the CICS product documentation. Yönergeleri kullanarak, Web hizmetini SOAP için WebSphere MQ iletimi için konuşlandırabilirsiniz.

İlgili görevler

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService
Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması
SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması
WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Configuring WebSphere Application Server to use W3C SOAP over JMS
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması
WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması

WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Başlamadan önce

Web hizmetini geliştirmek için Rational Application Developer, WebSphere Integration Developer ya da bir Web hizmetleri araç takımını kullanın.

Bu görev hakkında

WebSphere Application Server üzerinde SOAP iletimi olarak SOAP için WebSphere MQ iletimini kullanarak bir hizmeti konuşlandırmak üzere aşağıdaki yönergeleri kullanın.

Yordam

1. WebSphere MQ ' yu WebSphere Application Server 'da hizmet bütünleştirme veriyolu için JMS ileti alışverişi sağlayıcısı olarak yapılandırın.
2. Hizmetin gerektirdiği WebSphere MQ kaynaklarını yapılandırın.
3. WebSphere Application Server Network Deployment ürün belgelerinde Configuring JMS resources for the sync SOAP over JMS endpoint listenerbaşlıklı yönergeleri izleyin.
   Diğer WebSphere Application Server platformlarına ilişkin yönergeler vardır.
4. Hizmet URI 'sini, SOAP URI 'si için WebSphere MQ iletimiyle uyumlu olacak şekilde değiştirin.
5. Hizmeti WebSphere Application Server 'a konuşlandırın.

Sonraki adım

İstemcilerin hizmeti sorgulayıp WSDL ' yi yanıt olarak alabilmeleri için HTTP ile hizmeti bir iletim olarak konuşlandırın.

İlgili görevler

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService
Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması
SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması
SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

Configuring WebSphere Application Server to use W3C SOAP over JMS
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması
WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

Configuring WebSphere Application Server to use W3C SOAP over JMS

A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Başlamadan önce

Görev için WebSphere Application Server v7.0.0.9 ve WebSphere MQ v7.0.1.3gereklidir.

Bu görev hakkında

Görevin iki adımı vardır:

Yordam

1. “WebSphere MQ kaynaklarını yapılandır” sayfa 967
2. “WebSphere Application Server kaynaklarını yapılandır” sayfa 967

Sonraki adım

“WebSphere MQ kaynaklarını yapılandır” sayfa 967

İlgili görevler

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService
Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması
SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması
SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması
WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması
WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

WebSphere MQ kaynaklarını yapılandır

Başlamadan önce

For Axis2 support you required WebSphere MQ 7.0.1.3 or later.

Bu görev hakkında

Basitlik için, görev WebSphere MQ ' un diğer yazılımla aynı iş istasyonunda kurulu olduğunu varsayar ve bağ tanımları bağlantılarını kullanır. WebSphere Application Server ve Axis2 istemci yapılanışları istemci bağlantılarıyla çalışır. İstemci bağlantıları kullanarak görevle çalışmak için, hem Axis2 istemcisinden hem de WebSphere Application Server bilgisayarlarından gelen ve istek ve yanıt kuyruklarına ileti yerleştirebilir ve bu kuyruklara ileti alabilirsiniz.

Yine, basitlik için güvenlik yapılandırması kullanılmamaktadır. Kullanıcı kimliğinin tam `mqm` yetkisi vardır.

Yordam

1. Varsayılan bir kuyruk yöneticisi (`QM1`) yaratın.

   Varsayılan kuyruk yöneticisi olarak `QM1` yaratmak için WebSphere MQ Gezgini 'ni kullanın. Otomatik olarak başlayacak şekilde yapılandırın ve bir dinleyici oluşturmak için seçeneği belirleyin. Diğer bir seçenek olarak, aşağıdaki komutları kullanın:

   ```
   crtmqm -q -sa QM1
   strmqm
   echo define listener (LISTENER.TCP) trptype(tcp) ipaddr(localhost) port(1414)
               control(qmgr) replace | runmqsc
   echo start listener(LISTENER.TCP) | runmqsc
   ```

2. Bir istek kuyruğu (`REQUESTAXIS`) ve yanıt kuyruğu (`REPLYAXIS`) tanımlayın.

   Explorer aracını ya da aşağıdaki komutları kullanın:

   ```
   echo define ql(REQUESTAXIS) replace | runmqsc
   echo define ql(REPLYAXIS) replace | runmqsc
   ```

Sonraki adım

“WebSphere Application Server kaynaklarını yapılandır” sayfa 967

WebSphere Application Server kaynaklarını yapılandır

Başlamadan önce

For W3C SOAP over JMS support you require WebSphere Application Server v7. Bu yapılandırma, WebSphere Application Server Version 7.0 Test Environment v7.0.0.9 Update 1 üzerinde gerçekleştirildi. WebSphere Application Server , Rational Software Architect for WebSphere Software 7.5.4ile birlikte sağlanmıştır. Rational Software Architect was updated to v7.5.5.1, by applying the latest updates that were available.

Kuruluş işleminin bir parçası olarak, WebSphere Application Server için bir profil oluşturun. Görevdeki denetim güvenliği etkinleştirilmez. Varsayılan profil adı `was70profile1`olur ve sunucu `server1`' dir.

Bu görev hakkında

WebSphere Application Server 'ı yapılandırın. Sunucuyu Rational Application Developer 'dan başlatabilir ve Administrative konsolunu Servers (Sunucular) görünümünden başlatabilir ya da bir komut dosyasını kullanarak sunucuyu başlatabilir ve bir tarayıcı kullanarak sunucuyu yönetebilirsiniz. Görev ikinci yöntemi kullanır.

Sunucu komut dosyaları *Rational Installation Root*\SDP\runtimes\base_v7\profiles\was70profile1\binklasöründe yer alıyor. İncelenmek isteyebileceğiniz günlük dosyaları *Rational Installation Root*\SDP\runtimes\base_v7\profiles\was70profile1\logs\server1içinde.

Bir kural olarak, tüm WebSphere MQ nesne adları büyük harfle ve WebSphere MQ nesnelerine gönderme yapan tüm JNDI adları küçük harfli olur.

Yordam

1. Sunucuyu başlatın.

   ```
   startServer server1
   ```

2. Bir tarayıcı başlatın, yönetim konsolunu açın ve oturum açın.

   ```
   http://localhost:9061/ibm/console/unsecureLogon.jsp
   ```

   Kullanıcı kimliği alanına herhangi bir dizgi yazın.

3. Bir bağlantı üreticisi yaratın, `qm1`
   a) Gezginde, **Kaynaklar** > **JMS** > **Bağlantı fabrikaları**' yı açın.
   b) In the Connection factories window, select scope **Düğüm =*nodename***, click **Yeni**.
   c) **WebSphere MQ Messaging Provider** > **Tamam**öğelerini seçin.
   d) Provide the queue manager connection information from Çizelge 143 sayfa 968 > **Sonraki**.

   Çizelge 143. Kuyruk yöneticisi bağlantısı bilgileri

Alan adı	**Değer**
Ad	`qm1`
JNDI adı	`qm1`

   e) Bağlantı yöntemi olarak **Bu sihirbazda gereken tüm bilgileri girin** öğesini seçin > **Sonraki**.
   f) Kuyruk bağlantısı ayrıntıları olarak `QM1` yazın > **İleri**.
   g) Bağlantı ayrıntılarını Çizelge 144 sayfa 968 > **Sonraki**içinden girin.

   Çizelge 144. Bağlantı ayrıntıları

Alan adı	**Değer**
Aktarım	`Bindings, then client`
Anasistem adı	`localhost`
Kapı	`1414`
Sunucu bağlantı kanalı	`SYSTEM.DEF.SVRCONN`

   h) **Test bağlantısı** > **Sonraki** > **Son** > **Kaydet**seçeneklerini belirleyin.
4. JMS istek kuyruğunu yaratın, `requestaxis`.

a) Navigator'de, **Kaynaklar** > **JMS** > **Kuyruklar**' ı açın.

b) In the Connection factories window, select scope **Düğüm =*nodename***, click **Yeni**.

c) **WebSphere MQ Messaging Provider** > **Tamam**öğelerini seçin.

d) Enter the queue details from Çizelge 145 sayfa 969 > **Tamam** > **Kaydet**.

Çizelge 145. Kuyruk Ayrıntıları

Alan adı	**Değer**
Ad	`requestaxis`
JNDI adı	`requestaxis`
Kuyruk adı	`REQUESTAXIS`
Kuyruk yöneticisi adı	`QM1`

5. JMS yanıt kuyruğunu yaratmak için "4" sayfa 968 adımını yineleyin, `replyaxis`.
6. Bir etkinleştirme belirtimi oluşturun, `qm1as`.

   Etkinleştirme belirtimi, istek kuyruğuna bir ileti geldiğinde Web hizmeti yöneltici Message Driven Bean 'i (MDB) tetikler. MDB, Rational Application Developer Web hizmeti sihirbazı tarafından yaratılan Web hizmetine ilişkin konuşlandırma tanımlayıcısında tanımlanır.

   a) Navigator'de, **Kaynaklar** > **JMS** > **Etkinleştirme belirtimleri**' ne açın.

   b) In the Connection factories window, select scope **Düğüm =*nodename***, click **Yeni**.

   c) **WebSphere MQ Messaging Provider** > **Tamam**öğelerini seçin.

   d) Enter the basic attributes of the activation specification from Çizelge 146 sayfa 969 > **Sonraki**.

Çizelge 146. Etkinleştirme belirtimi adı

Alan adı	**Değer**
Ad	`qm1as`
JNDI adı	`qm1as`

   e) Specify its MDB information from Çizelge 147 sayfa 969 > **Sonraki**.

Çizelge 147. MDB bilgileri

Alan adı	**Değer**
Hedef JNDI adı	`requestaxis`
İleti seçici	*Sol boş*
Hedef tipi	`Queue`

   f) Bağlantı yöntemi olarak **Bu sihirbazda gereken tüm bilgileri girin** öğesini seçin > **Sonraki**.

   g) Kuyruk bağlantısı ayrıntıları olarak `QM1` yazın > **İleri**.

   h) Bağlantı ayrıntılarını Çizelge 144 sayfa 968 > **Sonraki**içinden girin.

Çizelge 148. Bağlantı ayrıntıları

Alan adı	**Değer**
Aktarım	`Bindings, then client`
Anasistem adı	`localhost`
Kapı	`1414`
Sunucu bağlantı kanalı	`SYSTEM.DEF.SVRCONN`

i) **Test bağlantısı** > **Sonraki** > **Son** > **Kaydet**seçeneklerini belirleyin.

7. Yanıt kuyruğu için bir kuyruk bağlantısı üreticisi (`jms/WebServicesReplyQCF`) yaratın.

   Web hizmetleri yönelticisi, bir yanıt kuyruğuna erişmek için bir kuyruk bağlantısı üreticisi kullanır. Web hizmetine ilişkin konuşlandırma tanımlayıcısında, kuyruk bağlantısı üreticisine varsayılan JNDI adı (`jms/WebServicesReplyQCF`) verilir. Konuşlandırma tanımlayıcısında adı değiştirebilirsiniz. Bu görevde, varsayılan adı JMS kaynak tanımlamalarına ekleyin.

   a) Navigator'de, **Kaynaklar** > **JMS** > **Kuyruk bağlantısı fabrikaları**' yı açın.

   b) In the Connection factories window, select scope **Düğüm =*nodename***, click **Yeni**.

   c) **WebSphere MQ Messaging Provider** > **Tamam**öğelerini seçin.

   d) Enter the basic attributes of the queue connection factory from Çizelge 149 sayfa 970 > **Sonraki**.

Çizelge 149. Kuyruk bağlantısı üreticisi adı	
Alan adı	**Değer**
Ad	`WebServicesReplyQCF`
JNDI adı	`jms/WebServicesReplyQCF`

   e) Bağlantı yöntemi olarak **Bu sihirbazda gereken tüm bilgileri girin** öğesini seçin > **Sonraki**.

   f) Kuyruk bağlantısı ayrıntıları olarak `QM1` yazın > **İleri**.

   g) Bağlantı ayrıntılarını Çizelge 144 sayfa 968 > **Sonraki**içinden girin.

Çizelge 150. Bağlantı ayrıntıları	
Alan adı	**Değer**
Aktarım	`Bindings, then client`
Anasistem adı	`localhost`
Kapı	`1414`
Sunucu bağlantı kanalı	`SYSTEM.DEF.SVRCONN`

   h) **Test bağlantısı** > **Sonraki** > **Son** > **Kaydet**seçeneklerini belirleyin.

Sonraki adım

"JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor" sayfa 929

Bir hizmetin WebSphere ESB ve Process Server hizmet uç noktasına WebSphere Transport for SOAP kullanacak şekilde konuşlandırılması

WebSphere MQ iletimi WebSphere ESB ve Process Servertarafından doğrudan desteklenmez. Özel bir Dışa Aktarma yapılandırmanız gerekir.

Bu görev hakkında

WebSphere Integration Developer, özel bir WebSphere MQ JMS SOAP Dışa Aktarma olanağı yaratmak için WebSphere MQ JMS Dışa Aktarma olanağına bağlayabileceğiniz bir SOAP veri dönüşümü sağlar.

SOAP için WebSphere MQ iletimi üzerinden SOAP isteklerini almak üzere özelleştirilmiş bir Dışa Aktarma yaratmak için yönergeleri izleyin.

Yordam

1. WebSphere Process Server for Multiplatforms V6.2 ürün belgelerinde Overview of exports and exports and How to connect to WebSphere MQ (WebSphere Process Server for Multiplatforms V6.2 ürün belgelerine bakın).

2. IBM Business Process Manager, Sürüm 8.6 ürün belgelerinde MQ JMS dışa aktarma bağ tanımı oluşturulması görevini izleyin.

   SOAP iletisini biçimlendirmek için Önceden paketlenmiş JMS veri biçimi dönüşümlerinde açıklanan SOAP veri bağ tanımını kullanın.

İlgili görevler

Deploying a service to Axis 1.4 to use for WebSphere transport for SOAP using amqwdeployWMQService
Bir konuşlandırma dizini yaratarak, **amqwdeployWMQService** komutunu çalıştırarak ve Axis 1.4 dinleyicisini başlatarak SOAP için bir Axis 1.4 hizmetini WebSphere MQ iletisine konuşlandırın.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 hizmetine bir hizmet konuşlandırılması
SOAP için bir .NET Framework 1 ya da 2 hizmetini WebSphere MQ iletimi için konuşlandırın. Bir konuşlandırma dizini yaratın, **amqwdeployWMQService** komutunu çalıştırın ve .NET dinleyicisini başlatın.

SOAP için WebSphere Transport 'u kullanmak üzere bir hizmet CICS Transaction Server 'a konuşlandırılması
SOAP içinWebSphere MQ iletimi CICS Transaction Server 4.1 Web hizmetleri desteğine tümleştirilmiştir.

WebSphere Transport for SOAP ' yi kullanmak için bir hizmetin WebSphere Application Server 'a konuşlandırılması
WebSphere MQ transport for SOAP, WebSphere Application Server 'da hizmet bütünleştirme veriyolu ile bütünleştirilmiştir.

Configuring WebSphere Application Server to use W3C SOAP over JMS
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 1 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application server using the W3C SOAP over JMS protocol. Configure the WebSphere MQ and WebSphere Application Server resources to develop and deploy Web service bound to W3C SOAP over JMS as a transport.

Web hizmeti istemcilerinin SOAP için WebSphere MQ iletimi kullanacak şekilde konuşlandırılması

Bir Web hizmeti istemcisini farklı istemci ortamlarından birine konuşlandırın ve SOAP için WebSphere MQ iletimi kullanarak bir hizmete bağlanın.

Başlamadan önce

Web hizmetini geliştirin ve SOAP için WebSphere MQ aktarımı olanağını kullanmak üzere bu hizmeti devreye alın.

Bu görev hakkında

Bir dizi farklı istemci ortamında SOAP için WebSphere MQ iletimi ile çalışmak üzere bir web hizmeti istemcisi konuşlandırabilirsiniz. You can deploy a Java client to Axis 1.4 using only the software installed with WebSphere MQ. Diğer istemci ortamları için ek yazılım kurmanız gerekir.

Devreye alma yönergeleri olan istemci ortamlarında SOAP için WebSphere iletimi çalıştırılmayla kısıtlanmadınız. Bir istemciyi desteklenen ortamlardan birine konuşlandırmak için yönergeleri kullanın.

Not: Some integrated environments offer SOAP over JMS using the W3C recommended JMS SOAP binding, as well as the WebSphere MQ transport for SOAP binding. Releases of WebSphere MQ, up to and including 7.0.1.2, support only the WebSphere MQ transport for SOAP binding. 7.0.1.3 ' tan başlayarak, JMS üzerinden SOAP için W3C aday önerisine uygun bir URI kullanarak Axis2 istemcilerini konuşlandırabilirsiniz. See the tutorial, WebSphere Application Server V7 ve Rational Application Developer V7.5ile SOAP/JMS JAX-WS Web hizmetleri uygulaması geliştirin.

Deploying a Web service client to Axis 1.4 to use IBM WebSphere MQ transport for SOAP

İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH' ı ayarlayın. IBM WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

Başlamadan önce

İpucu: Hizmeti HTTP ' ye konuşlandırın, HTTP için istemciyi geliştirin ve test edin ve daha sonra, SOAP için IBM WebSphere MQ iletimi için istemciyi değiştirin:

1. İstemciye Register.extension() çağrısını ekleyin.
2. SOAP için IBM WebSphere MQ iletimi için URI ' yi kullanmak üzere istemci yetkili sunucu konum belirleyici sınıfındaki statik web hizmeti adresini değiştirin.

Bu görev hakkında

SOAP için IBM WebSphere MQ iletimi kullanmak üzere bir Axis 1.4 istemcisinin konuşlandırılması, bir HTTP istemcisine kıyasla ek bir konuşlandırma adımı gerektirir. You must create a client deployment descriptor, `client-config.wsdd`, to map the `jms:` transport to the sender class `com.ibm.mq.soap.transport.jms.WMQSender`.

İstemci yetkili sunucuları oluşturmak için **amqwdeployWMQService** komutunu kullanırsanız, komutu oluşturulan dizinleri kullanarak istemciyi dağıtabilirsiniz.

Yordam

1. İstemci konuşlandırma dosyalarını tutmak için *deployDir* dizini yaratın.
2. Windows sistemlerinde bir komut penceresi açın ya da *deployDir*içinde UNIX and Linux sistemlerinde X Window System (X Pencere Sistemi) kullanarak bir komut kabuğu açın.
3. CLASSPATH' ı ayarlamak için **amqwsetcp.cmd** komutunu çalıştırın.
4. Axis 1.4 istemci konuşlandırma tanımlayıcısı yaratmak için **amqwclientconfig.cmd** komutunu, *deployDir*içinde `client-config.wsdd` komutunu çalıştırın.
5. İstemci paketindeki, istemci yetkili sunucu sınıflarındaki ve istemcinin kullandığı kitaplıkların CLASSPATH' da yer aldığından emin olun.

   amqwdeployWMQService , .NET istemci yetkili sunucularını `./generated/server/soap/client/remote/dotnetService` ve Axis 1.4 yetkili sunucularını `./generated/server/soap/client/remote/`*`client package`*' a yerleştirir.

Örnek

Örnek, bir Axis 1.4 Java istemcisinden yapılandırma ve çıkış (Şekil 199 sayfa 973) gösterir. İstemci, Şekil 198 sayfa 973, giriş parametresini yineleyen bir Web hizmetini çağırır. Hizmet tanımlaması (Şekil 197 sayfa 972), hizmet WSDL 'den alınan URI' yi gösterir.

```
<wsdl:service name="QuoteSOAPImplService">
    wsdl:port binding="intf:org.example.www.QuoteSOAPImplBindingSoap"
                      name="org.example.www.QuoteSOAPImpl_Wmq">
       <wsdlsoap:address location="jms:/queue?destination=REQUESTAXIS
           &connectionFactory=(connectQueueManager(QM1)binding(server))
           &initialContextFactory=com.ibm.mq.jms.Nojndi
           &targetService=org.example.www.QuoteSOAPImpl.java
           &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE" />
    </wsdl:port>
</wsdl:service>
```

Şekil 197. Hizmet tanımı

```
package org.example.www;
import com.ibm.mq.soap.Register;
public class QuoteClient {
   public static void main(String[] args) {
      try {
         Register.extension();
         QuoteSOAPImplServiceLocator locator = new QuoteSOAPImplServiceLocator();
         System.out.println("Response = "
                  + locator.getOrgExampleWwwQuoteSOAPImpl_Wmq().getQuote("IBM"));
      } catch (Exception e) {
         System.out.println("Exception = " + e.getMessage());
      }
   }
}
```

Şekil 198. Axis 1.4 Java istemcisi

```
C:\IBM\ID\Test>dir /s /b
C:\IBM\ID\Test\client-config.wsdd
C:\IBM\ID\Test\org
C:\IBM\ID\Test\org\example
C:\IBM\ID\Test\org\example\www
C:\IBM\ID\Test\org\example\www\GetQuoteFaultMsg.class
C:\IBM\ID\Test\org\example\www\OrgExampleWwwQuoteSOAPImplBindingSoapStub.class
C:\IBM\ID\Test\org\example\www\QuoteClient.class
C:\IBM\ID\Test\org\example\www\QuoteSOAPImpl.class
C:\IBM\ID\Test\org\example\www\QuoteSOAPImplService.class
C:\IBM\ID\Test\org\example\www\QuoteSOAPImplServiceLocator.class

C:\IBM\ID\Test>amqwsetcp
C:\IBM\ID\Test>java org.example.www.QuoteClient.class
Response = IBM
```

Şekil 199. İstemci yapılanışı ve çıkışı

Sonraki adım

1. İstemciyi bir IBM WebSphere MQ istemcisi olarak konuşlandırayorsanız, istemci ve sunucu bağlantı kanalını yapılandırın.
2. İstemciyi hizmete farklı bir kuyruk yöneticisine konuşlandırayorsanız, hedef kuyruğu istemci için kullanılabilir duruma getirmeniz gerekir. Hizmet kuyruğu yöneticisinden hedef kuyruğu bir küme kuyruğu olarak ya da istemci kuyruk yöneticisinden uzak kuyruk tanımı olarak yapılandırın.

İlgili görevler

SOAP için WebSphere MQ iletimi kullanmak üzere bir Web hizmeti istemcisinin Axis2 ' ye konuşlandırılması

İstemci için bir konuşlandırma dizini ve Axis2 yapılandırma dosyası hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH ' ı (CLASSPATH) ayarlayın. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma

A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 4 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application Server using the W3C SOAP over JMS protocol. Modify the URL in the Axis2 client developed for WebSphere MQ transport for SOAP to use the W3C candidate recommendation for SOAP over JMS.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ve 2 'ye Web hizmeti istemcisi konuşlandırılması

İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucusunu ve istemci sınıfını belirtin. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

SOAP için WebSphere MQ iletimi kullanmak üzere bir Web hizmeti istemcisinin Axis2 ' ye konuşlandırılması

İstemci için bir konuşlandırma dizini ve Axis2 yapılandırma dosyası hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH ' ı (CLASSPATH) ayarlayın. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

Başlamadan önce

İpucu: Hizmeti HTTP ' ye konuşlandırın. HTTP için istemciyi geliştirin ve test edin ve daha sonra, SOAP için WebSphere MQ iletimi kullanarak hizmete gönderme yapmak için URL ' yi değiştirin.

Bu görev, yönetilmeyen bir Axis2 istemcisinin Java Standard Edition' a nasıl konuşlandırılacağını gösterir. You might want to deploy an Axis2 client to a Web container. “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942uygulamasında, bir Web kapsayıcısında bir istemci geliştirdiniz ve bu istemciyi WebSphere Application Server Community Edition' a yerleştirdiniz. Sunucu yapılandırmasının bir parçası olarak, Axis2 yüzlemini etkinleştirdiniz ve Web taşıyıcısının yapılanışındaki kategoriyi de eklediniz. Diğer uygulama sunucularındaki Web kapsayıcılarını yapılandırmak için Axis2 belgelerine, http://ws.apache.org/axis2/1_4_1/installationguide.html#servlet_containerbelgesine ya da Web sunucusuyla birlikte sağlanan belgelere bakın.

Not: Axis2 terimi, sunucu uygulamacığı taşıyıcısını kullanır. Sunucu uygulamacığı taşıyıcısı, bir Web taşıyıcısıyla aynı.

Bu görev hakkında

SOAP için WebSphere MQ iletimi kullanmak üzere bir Axis2 istemcisinin konuşlandırılması, HTTP ' yi kullanmak için bir Axis2 istemcisinin konuşlandırılması gibi. Additional steps are required to provide a classpath to the WebSphere MQ JAR files, and to modify the Axis2 configuration file. Axis2 yapılandırma dosyası, JMS için ek bir giriş gerektirir. Giriş, JMS `transportSender`öğesini uygulayan SOAP JAR dosyası için WebSphere MQ iletimi anlamına gelir.

Axis2 provides a script, `axis2.bat` or `axis2.sh`, which simplifies client deployment; see the examples in Şekil 203 sayfa 976 and Şekil 204 sayfa 976.

Not:

1. `axis2.bat` ' in düzeltilmesi gereken bir hata var. The string `-Djava.ext.dirs="%AXIS2_HOME%\lib\"` must be changed to `-Djava.ext.dirs="%AXIS2_HOME%\lib\\"`.
2. `axis2.bat` ve `axis2.sh`' de, `-Djava.ext.dirs` , tüm Axis2 JAR dosyalarına, bunları sınıf yoluna ayrı olarak eklemek yerine hızlı bir şekilde gönderme olarak kullanılır. Ne yazık ki bu yaklaşım kusurlu, ve sadece bazı JRES ' ler ile çalışıyor. Bu, IBM JRE ' leriyle çalışmaz.

   JVM değiştirgesi `-Djava.ext.dirs="%AXIS2_HOME%\lib\\"`, Axis JAR dosyalarını JVM ' de kullanılabilir duruma getirir. JVM, Axis JAR dosyalarının bazılarını somutlaştırma girişiminde bulunur ve JVM ' ye bağlı olan ayrıntıları, bir hataya yol açar. Genellikle, yığın izlemesinde aşağıdaki satırlardan birini görebilirsiniz:

   ```
   org.apache.axiom.om.util.UUIDGenerator.getInitialUUID(UUIDGenerator.java:76)
   ```

   ya da `org.apache.axis2.deployment.DeploymentException: java.security.NoSuchAlgorithmException: MD5 MessageDigest not available`

   Yönetilmeyen bir Axis2 istemcisinin çalıştırılabilmenin doğru yolu, Axis2 JAR dosyalarının sınıf yoluna (classpath) eklenmesi. Sınıf yolu (classpath) yalnızca istemci uygulaması için kullanılabilir; JVM ' ye değil.

Yordam, yönetilmeyen bir Axis2 istemcisini axis2 komut dosyasını kullanmadan çalıştırmak için gereken genel adımları açıklar. The examples in Şekil 201 sayfa 976 and Şekil 202 sayfa 976 are scripts for Pencereler and Linux.

Yordam

1. Axis2 1.4.1 dosyasını http://ws.apache.org/axis2/download/1_4_1/download.cgi olanağından yükleyin ve bir klasöre (`Axis2-1.4.1`) açın.
2. `Axis2-1.4.1\conf`'ta `axis2.xml` ' u güncelleyin.
   a) `Axis2-1.4.1\conf`'ta `axis2.xml` ' u güncelleyin. Add WebSphere MQ transport for SOAP as a `transportSender`:

   ```
   <transportSender name="jms"
   class="com.ibm.mq.axis2.transport.jms.WMQJMSTransportSender"/>
   ```

   b) Gerekiyorsa, bağlantı havuzunun büyüklüğünü varsayılan değer olan 10 'un içinden değiştirin.

   ```
   <transportSender name="jms"
   class="com.ibm.mq.axis2.transport.jms.WMQJMSTransportSender">
   <parameter name="ResourcePoolCapacity">20</parameter>
   </transportSender>
   ```

   `ResourcePoolCapacity` , önbellekte kaç adet hizmet uç noktası girişi sakının tutulmaya devam olduğunu tanımlar. Değer en az 1 olmalıdır. Hizmet uç noktası girişlerinin sayısı önbellek büyüklüğünü aşarsa, yeni girişlere yer sağlamak için girdiler silinir. Uç nokta girişlerinin büyüklüğü değişir. Önbellek atılmasını önlemek için yeterli büyüklükte bir sayı ayarlayın.

   See step 3 in “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942.
3. Bir dizin oluşturun *deployDir*. Bu dizin altında, istemci ve istemci yetkili sunucularını içeren klasör yapısı kopyalanır. *deployDir* , bir Eclipse Java projesindeki *`project`*`\bin` klasörünün eşdeğeridir.
4. Open a command window on Pencereler, or a command shell using X Window System on UNIX and Linux systems, in *deployDir*.
5. Sınıf yolunu (classpath) yürürlükteki dizini (Axis2 JAR dosyaları, `com.ibm.mqjms.jar` ve `com.ibm.mq.axis2.jar`) içerecek şekilde güncelleyin.

   `com.ibm.mqjms.jar` , gerekli diğer tüm WebSphere MQ JAR dosyalarına başvurur.
6. İstemci programını başlatmak için **Java** komutunu kullanın.

Örnekler

Bir Axis2 istemcisi çalıştırmanın dört örneği, Şekil 202 sayfa 976 içinde Şekil 204 sayfa 976olarak listelenir. Şekil 200 sayfa 975 shows the output from running the asynchronous client listed in Şekil 183 sayfa 947.

```
cd C:\IBM\ID\Workspaces\Axis2docs\StockQuoteAxis2PojoClient\bin>
runpojo soap/client/SQA2AsyncClient

HTTP Sync: 55.25
Callback constructor
Waiting for HTTP callback
Result in Callback 55.25
HTTP poll: 55.25
JMS: Sync: 55.25
Callback constructor
Waiting for JMS callback
Result in Callback 55.25
JMS poll: 55.25
Press any key to continue . . .
```

Şekil 200. Output from running SQA2AsyncClient

```
@echo off
set AXIS2_HOME=C:\OpenSource\axis2-1.4.1
set JAVA_HOME=C:\IBM\Java50
set WMQ_HOME=C:\IBM\MQ\java\lib

setlocal EnableDelayedExpansion
set CLASSPATH=
set AXIS2_CLASS_PATH=
FOR %%c in ("%AXIS2_HOME%\lib\*.jar") DO set AXIS2_CLASS_PATH=!AXIS2_CLASS_PATH!;%%c

"%JAVA_HOME%\bin\java" -Daxis2.repo="%AXIS2_HOME%\repository"
-Daxis2.xml="%AXIS2_HOME%\conf\axis2.xml" -cp
".;%WMQ_HOME%\com.ibm.mqjms.jar;%WMQ_HOME%\com.ibm.mq.axis2.jar;%AXIS2_CLASS_PATH%" %1

pause
```

Şekil 201. runpojo.bat: Pencereler, using a classpath

```
export AXIS2_HOME=/home/OpenSource/axis2-1.4.1
export JAVA_HOME=/usr/lib/j2sdk1.5-ibm
# update classpath
AXIS2_CLASSPATH=""
for f in "$AXIS2_HOME"/lib/*.jar
do
  AXIS2_CLASSPATH="$AXIS2_CLASSPATH":$f
done
AXIS2_CLASSPATH="$AXIS2_HOME":"$JAVA_HOME/lib/tools.jar":"$AXIS2_CLASSPATH":"$CLASSPATH"
java -cp /home/alex/dev/sandbox/Soap/axis2:/opt/mqm/java/lib/com.ibm.mqjms.jar:
/opt/mqm/java/lib/com.ibm.mq.axis2.jar:$AXIS2_CLASSPATH
-Daxis2.xml=/home/alex/dev/sandbox/axis2-1.4.1/conf/axis2.xml %1
```

Şekil 202. runpojo.sh: Linux, sınıf yolu (classpath) kullanılıyor.

```
@echo off
set AXIS2_HOME=C:\OpenSource\axis2-1.4.1
set JAVA_HOME=C:\IBM\Java50
set WMQ_HOME=C:\IBM\MQ\java\lib

%AXIS2_HOME%\bin\axis2 -cp .;%WMQ_HOME%\com.ibm.mqjms.jar;%WMQ_HOME%\com.ibm.mq.axis2.jar; %1
pause
```

Şekil 203. runaxis2.bat: Pencereler, using axis2.bat

Not

```
export AXIS2_HOME=/home/OpenSource/axis2-1.4.1
export JAVA_HOME=/usr/lib/j2sdk1.5-ibm

%AXIS2_HOME%\bin\axis2 -cp .;%WMQ_HOME%\com.ibm.mqjms.jar;%WMQ_HOME%\com.ibm.mq.axis2.jar; %1
```

Şekil 204. runaxis2.sh: Linux, axis2.shkullanılarak

Not

İlgili görevler

Deploying a Web service client to Axis 1.4 to use IBM WebSphere MQ transport for SOAP
İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH' ı ayarlayın. IBM WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 4 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application Server using the W3C SOAP over JMS protocol. Modify

the URL in the Axis2 client developed for WebSphere MQ transport for SOAP to use the W3C candidate recommendation for SOAP over JMS.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ve 2 'ye Web hizmeti istemcisi konuşlandırılması
İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucusunu ve istemci sınıfını belirtin. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma

A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 4 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application Server using the W3C SOAP over JMS protocol. Modify the URL in the Axis2 client developed for WebSphere MQ transport for SOAP to use the W3C candidate recommendation for SOAP over JMS.

Başlamadan önce

You must first complete the task, “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942 to call **SimpleJavaListener** using an Axis2 client and the WebSphere MQ transport for SOAP protocol.

Ayrıca, önceki görevlerde Web hizmetini ve yapılandırılmış WebSphere MQ ve WebSphere Application Server ' i de yaratmış olmanız gerekir:

1. “WebSphere MQ kaynaklarını yapılandır” sayfa 967.
2. “WebSphere Application Server kaynaklarını yapılandır” sayfa 967.
3. “JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor” sayfa 929.

Görevdeki istemci Eclipse Galileo 'da çalışır. You might run the client from the command line by modifying the `Axis2.bat` file shipped with Axis2.

Bu görev hakkında

The only change you must make to the existing Axis2 `StockQuoteAxis` static client to call the StockQuoteAxis service hosted by WebSphere Application Server is to change the URL passed to the client. WSDL değiştirilmediği için, `soap.server` paketindeki yetkili sınıf sınıflarını kullanabilirsiniz.

İstemciye geçirilecek URL ' yi tanımlamaya ilişkin iki yaklaşımınız var. Oluşturulan `StockQuoteAxis.wsdl`'te olduğu gibi aynı URL' yi de kullanabilirsiniz. WebSphere Application Server JNDI dizinine erişmek için `jndiInitialContextFactory` ve `jndiURL` değiştirgelerinin eklenmesi gerekir. Başka bir yaklaşım da, URL ' yi değiştirmenin ve bir JNDI araması kullanılmadan istemciye `QM1`üzerindeki `REQUESTAXIS` ve `REPLYAXIS` kuyruklarına doğrudan erişimi vermesini sağlar.

Axis2 istemcisine geçirilen URL ' de tanımladığınız bağlantı değiştirgeleri, SOAP iletileri göndermek ve almak için gereken WebSphere MQ kuyruk yöneticisine ve kuyruklara bağlanmak için kullanılır. Axis2 istemcisine geçirilen bağlantı değiştirgeleri, hizmet tarafından kullanılmasını zorunlu kılmaz. You can use the distributed queuing capabilities of WebSphere MQ to decouple the client and service from using the same queue manager, or the same name server.

Yordam

1. Save the URL from the generated `StockQuoteAxis.wsdl` and close down Rational Application Developer to save on memory.

   Sunucu yapılandırmasını değiştirmediyseniz, Rational Application Developer kapatılırsa uygulama sunucusu durdurulur. Bu durumda, komut şu komutla başlar:

   ```
   startserver server1
   ```

2. Open Eclipse Galileo in the workspace with the Axis2 client project.

3. SQA2StaticClient.java' ı açın.

   Bkz. SQA2StaticClient.java.
4. URI ' nin queue değişkenini kullanarak hizmeti çağırın.
   a) URL adresini değiştirin.

      Yeni URI:

      ```
      jms:queue:REQUESTAXIS
                ?replyToName=REPLYAXIS
                &connectionFactory=connectQueueManager(QM1)Bind(Server)
                &targetService=StockQuoteAxis;
      ```

      Bunu StockQuoteAxis.wsdl' daki URL ile karşılaştırın:

      ```
      jms:jndi:requestaxis
               ?jndiConnectionFactoryName=qm1
               &targetService=StockQuoteAxis
      ```

      Şekil 205. StockQuoteAxis.wsdlURL 'si

      - REQUESTAXIS is now uppercase as it is a queue name and not a JNDI name.
      - QM1 ile bağlantı basit.
      - URI, Yanıtın gönderileceği yanıtının adını içermiyor. İstemci, yanıt beklediği kuyruğu tanımlamalıdır.

   b) Aynı **Çalıştır ...**öğesini kullanarak SQA2StaticClient.java komutunu çalıştırın. configuration as you did in the task, "Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse" sayfa 942.
5. URI 'yi adlandırma sunucusu olarak WebSphere Application Server kullanarak, URI' nin jndi değişkenini kullanarak hizmeti çağırın.
   a) Use the URL from StockQuoteAxis.wsdl, Şekil 205 sayfa 978, providing the missing parameters to use the naming service in WebSphere Application Server.

      Vermeniz gereken eksik parametreler ve değerler şunlardır:

Çizelge 151. Ek JNDI parametreleri		
Değiştirge	**Bu örnekte kullanılan değer**	**Tanım**
&jndiURL	iiop://localhost:2810 ya da corbaname:iiop:localhost:2810	Adlandırma sağlayıcısının URI 'si. WebSphere Application Server için varsayılan değer 2809 olarak ayarlanır. Bu, RMI bağlacının kapı numarası olarak da bilinir ve önyükleme kapısı olarak da bilinir. Değer, SystemOut.logiçinde listelenir. `00000000 NameServerImp A NMSV0018I: Name server available on bootstrap port 2810`
&jndiInitialContextFactory	com.ibm.websphere.naming. WsnInitialContextFactory	WebSphere Application Server tarafından kullanılan ilk bağlam üreticisinin adı.
&replyToName	replyaxis	REPLYAXIS kuyruğunun JNDI adı.

```
jms:jndi:requestaxis?
         &jndiURL=iiop//localhost:2810
         &jndiConnectionFactoryName=qm1
         &jndiInitialContextFactory=com.ibm.websphere.naming.WsnInitialContextFactory
         &targetService=StockQuoteAxis
         &replyToName=replyaxis;
```

b) JNDI araması için gereken JAR dosyalarını ekleyin.

Bu yapılandırmada, JMS URL 'sinin `jndi` değişkenini kullanarak görevi çalıştırmak için oluşturma yoluna aşağıdaki JAR dosyaları eklenmiştir:

- `com.ibm.jaxws.thinclient_7.0.0.jar` from *`Rational install directory`*`\SDP\runtimes\base_v7\runtimes`.
- `com.ibm.ws.runtime.jar` Kaynak: *`Rational install directory`*`\SDP\runtimes\base_v7\plugins`

Farklı bir JNDI sağlayıcısı için, farklı JAR dosyalarına gereksinim duyarsınız.

Oluşturma yolundaki diğer JAR dosyaları şunlardır:

i) *`WebSphere MQ Install directory`*`\java\lib`içindeki tüm JAR dosyaları.

ii) *`Axis2-1.5.1`*`\lib`içindeki tüm JAR dosyaları.

iii) Java 6.0 JRE.

c) Aynı **Çalıştır ...**öğesini kullanarak `SQA2StaticClient.java` komutunu çalıştırın. configuration as you did in the task, “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942.

Sonuçlar

Her iki durumda da, hizmet verdiği yanıt istemci konsolu görünümünde görüntülenir.

İlgili görevler

Deploying a Web service client to Axis 1.4 to use IBM WebSphere MQ transport for SOAP
İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH' ı ayarlayın. IBM WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

SOAP için WebSphere MQ iletimi kullanmak üzere bir Web hizmeti istemcisinin Axis2 ' ye konuşlandırılması
İstemci için bir konuşlandırma dizini ve Axis2 yapılandırma dosyası hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH ' ı (CLASSPATH) ayarlayın. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ve 2 'ye Web hizmeti istemcisi konuşlandırılması
İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucusunu ve istemci sınıfını belirtin. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ve 2 'ye Web hizmeti istemcisi konuşlandırılması

İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucusunu ve istemci sınıfını belirtin. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

Başlamadan önce

İpucu: Visual Studio kullanarak hizmeti ve istemciyi geliştirin ve test edin. Then modify the client for WebSphere MQ transport for SOAP.

1. .NET Framework 1 ya da 2 kullanarak bir hizmeti devreye alıyorsanız, hizmeti kitaplık olarak oluşturun (`.dll`). SOAP için WebSphere MQ iletimi kullanarak konuşlandırın.
2. İstemciye Register.Extension() çağrısını ekleyin.
3. *MQ_Install*`\bin`içinde yer alan amqsoap.dllöğesine bir başvuru ekleyin.
4. Change the static `Url Adresi` property in the client proxy class constructor to the `jms:/` URI, for WebSphere MQ transport for SOAP.

Bu görev hakkında

SOAP için WebSphere MQ iletimi kullanmak üzere .NET Framework 1 ya da 2 için bir Web hizmeti istemcisinin konuşlandırılması, ek bir konuşlandırma adımı gerektirir. `amqsoap.dll` kaydını .NET Framework ile kaydettirmeniz gerekir. `amqsoap.dll` is automatically registered as part of installing WebSphere MQ transport for SOAP, but you might need to register it again.

İstemci yetkili sunucuları oluşturmak için **amqwdeployWMQService** komutunu kullanırsanız, komutu oluşturulan dizinleri kullanarak istemciyi dağıtabilirsiniz.

Yordam

1. İstemci konuşlandırma dosyalarını tutmak için *deployDir* dizini yaratın.
2. *deployDir*içinde bir komut penceresi açın.
3. Hizmet Axis 1.4üzerinde çalıştırılacaksa, CLASSPATH ' ı ayarlamak için **amqwsetcp** komutunu çalıştırın.
4. Gerekiyorsa, .NET Framework ile `amqsoap.dll` kaydı için **amqwRegisterDotNet** komutunu çalıştırın.

Örnek

Bu örnek, bir .NET Framework V2 istemcisinden yapılandırma ve çıkış Şekil 208 sayfa 981' i gösterir. İstemci Şekil 207 sayfa 980, giriş parametresini yineleyen bir Web hizmetini çağırır. Statik Url tanımı (Şekil 206 sayfa 980), istemci yetkili sunucusunun oluşturucusunu gösterir.

```
public Quote() {
    this.Url = "jms:/queue?destination=REQUESTDOTNET
                &connectionFactory=(connectQueueManager(QM1)binding(server))
                &initialContextFactory=com.ibm.mq.jms.Nojndi
                &targetService=Quote.asmx
                &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE";
}
```

Şekil 206. Durağan istemci yetkili sunucu oluşturucusu

```
using System;
namespace QuoteClientProgram {
    class QuoteMain {
        static void Main(string[] args) {
            try {
                IBM.WMQSOAP.Register.Extension();
                Quote q = new Quote();
                Console.WriteLine("Response is: " + q.getQuote("ibm"));
            } catch (Exception e) {
                Console.WriteLine("Exception is: " + e);
            }
        }
    }
}
```

Şekil 207. İstemci programı

```
C:\IBM\ID\DotNet\QuoteClientProgram\QuoteClientProgram>dir /s /b
C:\IBM\ID\DotNet\QuoteClientProgram\QuoteClientProgram\QuoteClientProgram.exe
C:\IBM\ID\DotNet\QuoteClientProgram\QuoteClientProgram>quoteclientprogram
Response is: IBM
```

Şekil 208. Konfigürasyon ve çıkış

Sonraki adım

1. İstemciyi bir WebSphere MQ MQI istemcisi olarak konuşlandırayorsanız, istemci ve sunucu bağlantı kanalını yapılandırın.
2. İstemciyi hizmete farklı bir kuyruk yöneticisine konuşlandırayorsanız, hedef kuyruğu istemci için kullanılabilir duruma getirmeniz gerekir. Hizmet kuyruğu yöneticisinden hedef kuyruğu bir küme kuyruğu olarak ya da istemci kuyruk yöneticisinden uzak kuyruk tanımı olarak yapılandırın.

İlgili görevler

Deploying a Web service client to Axis 1.4 to use IBM WebSphere MQ transport for SOAP
İstemciye ilişkin bir konuşlandırma dizini ve konuşlandırma tanımlayıcısı hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH' ı ayarlayın. IBM WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

SOAP için WebSphere MQ iletimi kullanmak üzere bir Web hizmeti istemcisinin Axis2 ' ye konuşlandırılması
İstemci için bir konuşlandırma dizini ve Axis2 yapılandırma dosyası hazırlayın. İstemci yetkili sunucularını ve istemci sınıfını belirtin ve CLASSPATH ' ı (CLASSPATH) ayarlayın. WebSphere MQ kuyruklarını ve kanallarını yapılandırın, hizmeti başlatın ve istemciyi test edin.

JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma
A Web service bound to the W3C candidate recommendation for SOAP over JMS must run in the EJB container of a Java EE application server. This task is step 4 of connecting an Axis2 Web service client and a Web service deployed to WebSphere Application Server using the W3C SOAP over JMS protocol. Modify the URL in the Axis2 client developed for WebSphere MQ transport for SOAP to use the W3C candidate recommendation for SOAP over JMS.

Bir Axis2 istemcisini JMS ve WebSphere Application Server üzerinden W3C SOAP olanağını kullanarak bir JAX-WS hizmetine bağlayın

Bu görevi tamamladığınızda, bir Axis2 istemcisinden WebSphere Application Server 'da çalışan bir JAX-WS Web hizmeti (JAX-WS Web hizmeti) olarak adlanmanıza neden olur. Axis2 istemcisi ve WebSphere Application Server, WebSphere MQüzerinde çalışan SOAP over JMS iletişim kuralı için W3C aday önerisini kullanır. Sırasıyla Web hizmeti istemcisi ve Web hizmeti oluşturmak için Eclipse Galileo ve Rational Application Developer olanağını kullanın.

Başlamadan önce

Bu görev için Rational Software Development Environment ve WebSphere Application Server sürüm 7 gereklidir. The task was created using the Rational Application Developer packaged with Rational Software Architect for WebSphere Software v7.5.5.1, and WebSphere Application Server Version 7.0 Test Environment v7.0.0.9 Update 1. Ayrıca, WebSphere MQ v7.0.1.3için de gereklidir.

Görev, “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922ve “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942diğer iki görev üzerinde oluşturulur. Bu görevleri tamamlamak için geliştirme ortamınızda Eclipse Galileo, WASCE, WASCE için Eclipse eklentisi ve Axis2 1.4.1 kurulu olmalıdır. Bu görev için WASCE ' yi zorunlu kılmayın.

Bazı adımlar karmaşıkmış. Bu adımlar, RationalRational Application Developer kullanan WebSphere Application Server için Web hizmeti uygulamalarını geliştirmekte olan bazı benzerliği kabul eder.

Görevin işlemci ve bellek gereksinimleri büyük olur. Görev, VMWare Pencereler XP SP3 sanal makinesinde,memorybelleğindeki 1.8GB sanal makinesinde gerçekleştirilmişti.

Görevi başlatmadan önce tüm yazılımı kurun. Yazılım, manyetik bant genişliğine bağlı olarak, karşıdan yüklemek ve kurmak için bir gün alır. Görev en az yarım gün sürer.

Bu görev hakkında

Bu görevin senaryosu, açık kaynaklı bir aracı (Eclipse Galileo) kullanarak bir hisse senedi (quote) web hizmeti (`StockQuoteAxis`) geliştirmiş olmadığınız bir senaryodur. `StockQuoteAxis` , WASCE açık bir kaynak sunucuda çalışan HTTP üzerinden SOAP kullanılarak konuşlandırılır.

Konuşlandırdığınız Web hizmetlerini, SOAP over JMS ya da Web hizmetleri güvenilir ileti sistemi gibi standartlara dayalı bir ileti iletimi ile HTTP üzerinden SOAP ' ya bağlamak istiyorsunuz. Hem istemci, hem de hizmet, standartlara dayalı arabirimler kullanmasını istiyorsunuz. Bu nedenle, gelecekteki projelerinizin geliştirme ekibi, SOAP için WebSphere MQ aktarımı kullanarak bir çözüm uygulamış olsa da, üretime geçmediniz.

The Axis2 client has removed the problem that the SOAP client for the WebSphere MQ transport for SOAP required a change from the HTTP client. Sorun, SOAP için IBM WebSphere MQ iletimi ile bağlı hizmetin, WebSphere MQ: `SimpleJavaListener`tarafından sağlanan özel bir dinleyici tarafından barındırıldığı hala devam ediyor.

With the W3C SOAP over JMS standard in candidate recommendation status, some vendors are providing support for W3C SOAP over JMS. Destek, bir Web hizmetini bir uygulama sunucusuna konuşlandırmanızı ve çeşitli bağlanırlık protokollerini kullanarak aynı hizmete bağlanmanızı sağlar. WebSphere Application Server v7 tarafından sağlanan destek, ileti tabanlı bir SOAP iletimi kullanabilmek için Web hizmetini ayrı olarak barınabilmek için gereken sorunu ortadan kaldırır. Standartlara dayalı ileti taşıma arabirimi, JMS kullanımı, farklı satıcılardan gelen araçları kullanarak çözüm geliştirebileceğiniz anlamına gelir. You hope the Web services tools in Eclipse will include SOAP over JMS bindings in the future.

Adımların çoğu Eclipsekullanılarak ya da WebSphere ürünleriyle birlikte sağlanan yönetim araçları kullanılarak gerçekleştirilir. Adımlar, bir Windows ortamı için açıklanmaktadır. Bazı komutlarda hafif değişiklikler yapılması ile diğer platformlardaki adımları gerçekleştirebilirsiniz.

The preliminary steps creating the HTTP Web service, and connecting to it using Axis2 are listed. Çözüm oluşturmak için bu adımlardan istemci ve WSDL kullanılır.

Yordam

1. SOAP için Axis2 istemcisi ve IBM WebSphere MQ iletimi kullanan StockQuoteAxis Web hizmetine bağlan
   a) “Developing a JAX-RPC service for WebSphere MQ transport for SOAP using Eclipse” sayfa 922
   b) “Developing a JAX-WS client for WebSphere transport for SOAP using Eclipse” sayfa 942
   c) “SOAP için WebSphere MQ iletimi kullanmak üzere bir Web hizmeti istemcisinin Axis2 ' ye konuşlandırılması” sayfa 974
2. Bir Axis2 istemcisi kullanarak StockQuoteEkseni Web hizmetine ve JMS üzerinden SOAP için W3C aday önerisine bağlanın.
   a) “WebSphere MQ kaynaklarını yapılandır” sayfa 967
   b) “WebSphere Application Server kaynaklarını yapılandır” sayfa 967
   c) “JMS üzerinde W3C SOAP için JAX-WS EJB Web hizmeti geliştiriyor” sayfa 929
   d) “JMS üzerinden W3C SOAP kullanarak bir Axis2 istemcisine konuşlandırma” sayfa 977

HTTP içinWebSphere MQ köprüsü

With WebSphere MQ bridge for HTTP, client applications can exchange messages with WebSphere MQ without the need to install a WebSphere MQ MQI client. You can call WebSphere MQ from any platform or language with HTTP capabilities.

HTTP için WebSphere MQ köprüsünün tanıtımı

HTTP için WebSphere MQ köprüsü bir Java, Enterprise Environment (JEE) Web uygulamasıdır. HTTP clients can send **POST**, **GET**, and **DELETE** requests to it to put, browse and delete messages from WebSphere MQ queues. HTTP için WebSphere MQ köprüsü, iletilerle birlikte kullanım için uygun değilse, teslim edilmesi gerekliyse.

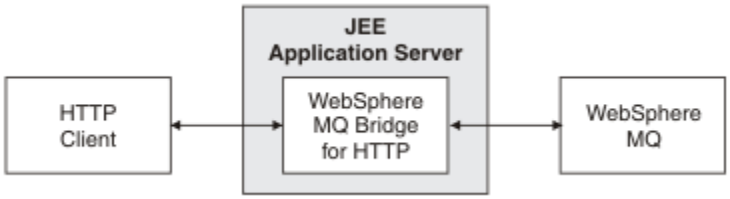
Yararlar

HTTP için WebSphere MQ köprüsüyle, çok çeşitli ortamlardan HTTP kullanarak WebSphere MQ iletileri gönderebilir ve alabilirsiniz:

- HTTP 'yi destekleyen, ancak WebSphere MQ' u destekleyen ortamlar.
- Bir WebSphere MQ MQI istemcisi kurmak için yeterli depolama alanı olmayan ortamlar.
- WebSphere MQerişimi gerektiren her sisteme WebSphere MQ MQI istemcisini kurmak için çok sayıda olan ortamlar.
- Web-based applications from which you want to send or receive messages without coding your own bridge to WebSphere MQ.
- AJAX gibi zamanuyumsuz teknikler kullanarak, geliştirmek istediğiniz web tabanlı uygulamalar. WebSphere MQ bridge for HTTP makes WebSphere MQ queues and topics available using Representation State Transfer (REST) over HTTP.

HTTP desteği, her iki noktadan noktaya iletişim ve yayınlama/abone olma ileti sistemi topolojileri ile kullanılabilir.

HTTP desteği nasıl çalışır?



Şekil 209. HTTP içinWebSphere MQ köprüsü

HTTP Web uygulaması için WebSphere MQ köprüsü bir ya da daha çok istemciden gelen HTTP isteklerini alır. Adlarında WebSphere MQ ile etkileşimde bulunur ve bunlara HTTP yanıtları döndürür.

HTTP için WebSphere MQ köprüsü, bir kaynak bağdaştırıcısı kullanarak WebSphere MQ ' ya bağlı bir JEE sunucu uygulamasıdır. The HTTP servlet handles three different types of HTTP requests: **POST**, **GET**, and **DELETE**.

Çizelge 152. HTTP fiilleri için WebSphere MQ köprüsü

HTTP İsteği	**Sonuç**
POST	Bir kuyruğa ya da konuya ileti koyar.
GET	Kuyruğun ilk iletisine göz atar. HTTP protokolü doğrultusunda, **GET** kuyruktan iletiyi silmez. Yayınlama/abone olma ileti alışverişi ile **GET** kullanmayın.
SİL	Bir kuyruktan ya da konudan bir iletiyi alır ve siler.

HTTP POST örneği

HTTP **POST** , bir iletiyi kuyruğa koyar ya da bir konuya yayın yapar. **HTTPPOST** Java örneği, bir iletinin kuyruğa ilişkin HTTP **POST** isteğinin bir örneğidir. Java kullanmak yerine, bir tarayıcı formu ya da AJAX araç takımı kullanarak bir HTTP **POST** isteği yaratabilirsiniz.

Şekil 210 sayfa 984 , myQueueadlı kuyruğa ileti koymak için bir HTTP isteği gösterir. Bu istek, WebSphere MQ iletisinin ilinti tanıtıcısını ayarlamak için x-msg-correlId HTTP üstbilgisini içerir.

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

Here is my message body that is posted on the queue.
```

*Şekil 210. Bir kuyruğa ilişkin HTTP **POST** isteği örneği*

Şekil 211 sayfa 984 , istemciye gönderilen yanıtı gösterir. Yanıt içeriği yok.

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

Şekil 211. HTTP POST yanıtına ilişkin örnek

HTTP DELETE örneği

HTTP **DELETE** , kuyruktan bir ileti alır ve iletiyi siler ya da bir yayını alır ve siler. **HTTPDELETE** Java örneği, bir kuyruktan ileti okuma isteğinde bulunan bir HTTP **DELETE** örneğidir. Java kullanmak yerine, bir tarayıcı formu ya da AJAX araç takımı kullanarak bir HTTP **DELETE** isteği yaratabilirsiniz.

Şekil 212 sayfa 984 is an HTTP request to delete the next message on queue called myQueue. Yanıtta, ileti gövdesi istemciye döndürülür. WebSphere MQ terimlerinde, HTTP **DELETE** yıkıcı bir alma koşuldur.

The request contains the HTTP request header x-msg-wait, which instructs WebSphere MQ bridge for HTTP how long to wait for a message to arrive on the queue. İstek, istemcinin yanıttaki ileti ilintilendirme tanıtıcısını almak olduğunu belirten x-msg-require-headers istek üstbilgisini de içerir.

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

*Şekil 212. HTTP **DELETE** isteği örneği*

Şekil 213 sayfa 984, istemciye verilen yanıttır. İlinti tanıtıcısı, isteğin x-msg-require-üstbilgileri içinde istendiği şekilde istemciye döndürülür.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that is retrieved from the queue.
```

*Şekil 213. HTTP **DELETE** yanıtı örneği*

HTTP GET örneği

HTTP **GET** , bir kuyruktan ileti alır. İleti kuyruğun üzerinde kalır. WebSphere MQ terimlerinde, HTTP **GET** bir göz atma isteğidir. Bir Java istemcisi, bir tarayıcı formu ya da AJAX araç takımı kullanarak bir HTTP **GET** isteği yaratabilirsiniz.

Şekil 214 sayfa 985 is an HTTP request to browse the next message on queue called `myQueue`.

The request contains the HTTP request header `x-msg-wait`, which instructs WebSphere MQ bridge for HTTP how long to wait for a message to arrive on the queue. İstek, istemcinin yanıttaki ileti ilintilendirme tanıtıcısını almak olduğunu belirten `x-msg-require-headers` istek üstbilgisini de içerir.

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

*Şekil 214. HTTP **GET** isteği örneği*

Şekil 215 sayfa 985 , istemciye döndürülen yanıdır. İlinti tanıtıcısı, isteğin `x-msg-require-üstbilgileri` içinde istendiği şekilde istemciye döndürülür.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that appears on the queue.
```

*Şekil 215. HTTP **GET** yanıtı örneği*

HTTP için WebSphere MQ köprüsünü kurma, yapılandırma ve doğrulama

"Java İleti Sistemi ve Web Hizmetleri" ' u WebSphere MQ MQI istemcisi ya da sunucu kuruluş malzemelerinden kurarak HTTP için WebSphere MQ köprüsünü edinin. HTTP için WebSphere MQ köprüsünü uygun bir uygulama sunucusuna konuşlandırın.

Başlamadan önce

Önkoşul olan ürünleri IBM WebSphere MQiçin Sistem Gereksinimleriadresinden denetleyin. Kuruluş işlemi, HTTP için WebSphere MQ köprüsünü çalıştırmak için önkoşul olan yazılımların varlığını ve kullanılabilirliğini denetmiyor. Önkoşulların kurulu olduğunu doğrulamanız gerekir.

HTTP için WebSphere MQ köprüsü, WebSphere MQ kaynak bağdaştırıcısını kurarak herhangi bir Java EE 1.4 uyumlu uygulama sunucusu üzerinde çalışır. You can also run WebSphere MQ bridge for HTTP on a WebSphere Application Server release earlier than version 6.0.2.1. JMS sağlayıcısı olarak WebSphere MQ ' u bütünleştirmek için WebSphere Application Server Message Listener Port (MLP) olanağını kullanın.

HTTP için WebSphere MQ köprüsü desteği yalnızca aşağıdaki uygulama sunucuları için sağlanır:

- WebSphere Application Server 6.0.2.1 ve sonraki sürümü.
- WebSphere Application Server Community Edition Sürüm 1.1 ve sonraki yayın düzeyi.

Bu görev hakkında

HTTP içinWebSphere MQ köprüsü `.war` dosyası olarak sağlanır. `WMQHTTP.war`.

- UNIX altyapılarında ve Linux,

- `WMQHTTP.war` , "Java Messaging and Web Services" kuruluş seçeneğinin bir parçası olarak dahil edilir. Bu seçenek hem istemci, hem de sunucu kuruluş malzemelerinde kullanılabilir.
- `WMQHTTP.war` , `<mqmtop>/java/http/WMQHTTP.war`dizinine kurulur. < *mqmtop*> , WebSphere MQ ' un kurulu olduğu dizindir.
- `WMQHTTP.samples` , `<mqmtop>/java/http/samples`dizinine kurulur. < *mqmtop*> , WebSphere MQ ' un kurulu olduğu dizindir.

Carry out the following installation steps to install WebSphere MQ bridge for HTTP, deploy, and configure it, and verify the configuration. Yapılandırma adımlarının ayrıntıları farklı uygulama sunucularına göre değişir. Use “Deploying and verifying WebSphere MQ bridge for HTTP on WebSphere Application Server V6.1.0.9” sayfa 986 as a template for the steps to follow on your application server.

Yordam

1. WebSphere MQ MQI istemcisini ya da sunucusunu kurarak `WMQHTTP.war` ' i edinin.
2. `WMQHTTP.war` dosyasını bir uygulama sunucusunda konuşlandırılabildiği bir sunucuya kopyalayın.
3. `WMQHTTP.war` uygulamasını bir uygulama sunucusuna konuşlandırın.
4. Gerekiyorsa, uygulama sunucunuzda bir kaynak bağdaştırıcısı olarak WebSphere MQ ' yı kurun.

   WebSphere MQ ' un uygulama sunucunuzda ileti alışverişi sağlayıcısı olarak önceden yapılandırılmış olup olmadığını öğrenin. WebSphere MQaramak için uygulama sunucunuzla birlikte sağlanan yönetim ya da yönetim aracını kullanın. WebSphere MQ aşağıdaki yol altında bulunabilir: **Kaynaklar > JMS > İleti alışverişi sağlayıcıları**.
5. Uygulama sunucusunda, WebSphere MQ MQI istemci aktarımı kullanan bir kuyruk yöneticisine bağlanmak için bir bağlantı üreticisi yapılandırın[12].
6. Bağlantı üreticisini kullanmak için uygulama sunucusunda `WMQHTTP.war` Web uygulamasını yapılandırın
7. Yapılandırmayı doğrulayın.
   a) Bağlantı fabrikada ve yerel bir kuyrukta adı belirtilen kuyruk yöneticisini ayarlayın.
   b) Yerel kuyruğa ileti yerleştirin.
   c) Yerel kuyruğa okuma ve yazma yetkisi bulunan, bağlantı üreticisinde adı belirtilen sunucu bağlantısı kanalını yaratın.
   d) Kuyruk yöneticisini ve dinleyiciyi başlatın.
   e) Uygulama sunucusunu ve `WMQHTTP.war`çalıştırılmamışsa, başlatın.
   f) Bir tarayıcı açın ve `http://`*`hostname`*`:`*`web port`*`/`*`Context root`*`/msg/queue/`*`local queue`*yazın

Sonuçlar

Tarayıcı penceresi, yerel kuyruğa yerleştirdiğiniz iletiyi görüntüler.

Sonraki adım

1. Örneği (“Deploying and verifying WebSphere MQ bridge for HTTP on WebSphere Application Server V6.1.0.9” sayfa 986) deneyin.
2. Örnek HTTP Java uygulamalarını çalıştırın.

Deploying and verifying WebSphere MQ bridge for HTTP on WebSphere Application Server V6.1.0.9

Örnek HTTP Java programlarını çalıştırmak üzere HTTP için WebSphere MQ köprüsünün konuşlandırılmasını hazırlamak için aşağıdaki örneği kullanın. Konuşlandırma, WebSphere Application Server V6.1.0.9' da yer alıyor.

[12] Başlangıçta, en azından istemci aktarımı yapılandırılıyor. Bazı uygulama sunucuları, doğrudan ya da bağ tanımları kipi bağlantıları kullanarak WebSphere MQ ' ya bağlanabilir.

Başlamadan önce

1. `WMQHTTP.war` dosyasını, WebSphere Application Server kurulumunuz için erişilebilir bir sunucuya kopyalamak için “HTTP için WebSphere MQ köprüsünü kurma, yapılandırma ve doğrulama” sayfa 985içindeki yönergeleri izleyin.
2. Yapılandırmayı sınamak üzere kullanmak için kuyruk yöneticisi ve kuyruk yapılandırın:
   - Örnekte, kuyruk yöneticisi Çizelge 153 sayfa 987içindeki değerleri kullanacak şekilde yapılandırılır:

Çizelge 153. Kuyruk yöneticisi yapılanışı	
Nesne	**Değer**
Anasistem adı	`itso-01`
Kuyruk yöneticisi	`QM1`
Yerel kuyruk	`HTTPTESTQ`
Sunucu bağlantı kanalı	`MYSVRCON`. Configure an MCA user ID with sufficient authority to read and write to `HTTPTESTQ`.
Dinleyici kapısı	`1414`

3. Kuyruk yöneticisini ve dinleyiciyi başlat
4. Bir sınama iletisini `HTTPTESTQ`' e yerleştirin. Örneğin:
   a. WebSphere MQ Explorer 'ı başlatın.
   b. QM1için yerel kuyruklar listesinde **HTTPTESTQ > Sınama iletisi koy > tip `First Message` > İleti koy > Kapat**simgesini sağ tıklatın.
5. Uygulama sunucusunu başlatın ve Integrated Solutions Console (Tümleşik Çözümler Konsolu) üzerinde oturum açın.

Bu görev hakkında

Örnek, uygulama sunucunuz olarak WebSphere Application Server V6.1.0.9 ' u çalıştırıyorsanız, atılacak adımları gösterir. Farklı bir WebSphere Application Server sürümü çalıştırıyorsanız ya da farklı bir uygulama sunucusu çalıştırıyorsanız, adımlar farklı olur. WebSphere Application Server V6.1.0.9 is pre-configured with WebSphere MQ installed as a message provider, using the WebSphere MQ MQI client libraries. WebSphere MQ bir ileti alışverişi sağlayıcısı olarak önceden yapılandırılmış değilse ya da WebSphere MQ sunucusu bağ tanımlarını kullanmak istiyorsanız, JEE için WebSphere MQ kaynak bağdaştırıcısını uygulama sunucunuza kurmanız ve yapılandırmanız gerekir.

Follow the instructions to deploy WebSphere MQ bridge for HTTP onto WebSphere Application Server V6.1.0.9, and verify the deployment using a browser:

Yordam

1. Gezinme bölmesinde **Kaynaklar > JMS sağlayıcıları > WebSphere MQ Messaging Provider**seçeneklerini tıklatın.

   WebSphere Application Server konuşlandırmasına bağlı olarak, Düğüm, Hücre ya da Sunucu düzeyinde yapılandırabilirsiniz. Örnek, Sunucu düzeyinde devreye alma olanağını kullanır.
2. **Ek özellikler**altında **Bağlantı fabrikaları > Yeni**öğelerini tıklatın.
3. JMS sağlayıcılar formunda, Çizelge 154 sayfa 987içindeki bilgileri ya da seçtiğiniz alternatifleri belirtin ve **Uygula > Kaydet**seçeneğini tıklatın.

Çizelge 154. Aşağıdaki alanları ayarlayın ya da değiştirin	
Alan	**Değer**
Ad	`WMQHTTPBridge`

Çizelge 154. Aşağıdaki alanları ayarlayın ya da değiştirin (devamı var)	
Alan	**Değer**
JNDI adı	`jms/WMQHTTPJCAConnectionFactory`
Kuyruk yöneticisi	`QM1`
Anasistem	`itso-01`
Kapı	`1414`
Kanal	`MYSVRCON`
İletim tipi	`CLIENT`

4. Gezinme bölmesinde **Applications > Install New Application**(Uygulamalar > Yeni Uygulama Kur) öğelerini tıklatın.
5. Yolu `WMQHTTP.war` 'e ekleyin ve bir bağlam kökü sağlayın, **İleri**' yi tıklatın.
   a) Bağlam kökü isteğe bağlıdır. `mq` , örnek HTTP uygulamaları için varsayılan bağlam köküdür.
   b) Bağlam kökü, HTTP için WebSphere MQ köprüsünü tanımlayan URI ' nin bir kısmını oluşturur. Bağlam kökünü atlayabilir ya da daha sonra değiştirebilirsiniz.
6. Kuruluş sihirbazının **Kuruluş seçeneklerini belirle** sayfasında varsayılan değerlerin hiçbirini değiştirmenize gerek yoktur, **İleri**düğmesini tıklatın.
7. **Birimleri sunucularla eşle** sayfasında, bir Küme ya da Sunucu seçin, Seç kutusunu işaretleyin, **Uygula > İleri**düğmesini tıklatın.
8. **Kaynak başvurularını kaynak eşle** sayfasında, **javax.jms.ConnectionFactory** formunda **Göz At ...**düğmesini tıklatın. HTTP satırı için IBM WebSphere MQ köprüsünde.
9. **Enterprise Applications > Available resourcalar** sayfasında **WMQHTTPBridge**seçeneğini belirleyin ve **Apply**(Uygula) düğmesini tıklatın.
10. **javax.jms.ConnectionFactory** formunu geri almak için, kimlik doğrulama yöntemini seçin.
    a) Örneğin, **Yok**'u seçin, **Uygula**' yı tıklatın. Diğer seçenekler ek yapılandırma gerektirir.
11. HTTP için IBM WebSphere MQ köprüsü için **Seç** onay kutusunu işaretleyin, **İleri > Sonraki > Son > Kaydet**seçeneklerini tıklatın.
12. Gezinme bölmesinde **Applications > Enterprise Applications**(Uygulamalar > Kurumsal Uygulamalar) öğelerini tıklatın.
13. `WMQHTTP.war`için seçim kutusunu işaretleyin, **Başlat**' ı tıklatın.
14. Bir tarayıcı penceresi açın. Uygun anasistem adı ve bağlantı noktasını kullanarak `http://itso-01:9080/mq/msg/queue/HTTPTESTQ`yazın.

Sonuçlar

Yapılandırma başarılı olursa, tarayıcı penceresi `First Message`(sayfa.) görüntülenir.

Sonraki adım

Örnek HTTP Java uygulamalarını çalıştırın.

HTTP için WebSphere MQ köprüsünü kullanarak yayınlama/abone olma

HTTP içinWebSphere MQ köprüsü, JMS yayınlama/abone olma arabirimi için WebSphere MQ sınıflarını kullanır. HTTP **POST** bir yayın yaratır. HTTP **DELETE** , dayanıklı olmayan bir yönetilen abonelik yaratır. Konu URI 'sını kullanmadan önce, JMS için yayınlama/abone olma özelliğini yapılandırmalısınız.

Yayınlama/abone olma, sürüm 7 'deki WebSphere MQ ' a tam olarak tümleştirilmiştir. Sürüm 7 'den önce, yayınları ve abonelikleri ayrı bir yayınlama/abone olma aracısı olarak ele alınır. Sürüm 7 'deki tam bütünleştirilmiş yayınlama/abone olma ile ayırt etmek için "kuyruğa alındı" yayınlama/abone olma olarak adlandırılır. Sürüm 7, tümleşik yayınlama/abone olma özelliğini kullanarak kuyruğa yollanmış yayınlama

aboneliğini öykünmektedir. Öykünme, aynı kuyruk yöneticisiyle çalışan tümleşik uygulamalarla birlikte var olan kuyruğa alınmış yayınlama/abone olma uygulamalarının birlikte var olması için olanak sağlar. Kuyruğa alınan yayınlama/abone olma uygulamaları aynı konuları paylaşır ve tümleşik uygulamalarla da çalışabilir. Sürüm 6 'da, aracı WebSphere MQile birlikte gönderilmişti; sürüm 6 'dan önce SupportPackolarak kullanılabilir.

Yapılandırma

HTTP için WebSphere MQ köprüsü, yayınlamak ve abone olmak için JMS arabirimini kullanır. Sürüm 7 'de, JMS için WebSphere MQ sınıflarının kuyruğa alınmış ya da tümleşik yayınlama/abone olma, `PROVIDERVERSION` JMS özelliğini kullanarak mı kullanacağınızı denetleyebilirsiniz.

An additional consideration is that you can use either WebSphere MQ MQI client libraries with WebSphere MQ bridge for HTTP, or server libraries. Sürüm 6 istemci kitaplıkları, yalnızca kuyruğa alınan yayınlama/abone olma özelliğini destekler; sürüm 7 kitaplıkları kuyruğa alınmış ve tümleşik yayınlama/abone olma özelliğini destekler. Most Web or application servers that use WebSphere MQ as a messaging provider do so using client libraries. Tümleşik yayınlama/abone olma aboneliğini kullanmak için, hem WebSphere MQ MQI istemcisi ve sunucu kitaplıklarının en az 7. sürümünde olması gerekir. WebSphere ' in önceki bir sürümünü 7 'den önce çalıştırıyorsanız, kuyruğa alınmış yayınlama/abone olma özelliğini yapılandırmanız gerekir; bkz. Çizelge 155 sayfa 989. Kullandığınız Web sunucusu ya da uygulama sunucusu ile hangi kitaplıkların kurulduğunu ya da yapılandırıladığınızı denetleyin.

Çizelge 155. Yayınlama/abone olma yapılandırma kipleri

	İstemci V6 ya da önceki yayın düzeyi	**İstemci V7 ya da üstü**
Sunucu V6 ya da önceki yayın düzeyi	1. `\java\bin\MQJMS_PSQ.mqsc` komut dosyasını çalıştır	Desteklenmiyor
Sunucu V7 ya da üstü	1. `\java\bin\MQJMS_PSQ.mqsc` komut dosyasını çalıştır 2. Kuyruk yöneticisini `PSMODE=ENABLED`olarak ayarlayın.	1. PROVIDERVERSION = 7 ' DE a. Kuyruk yöneticisini `PSMODE=ENABLED` ya da `PSMODE=COMPAT`olarak ayarlayın. 2. PROVIDERVERSION = 6 ise a. Kuyruk yöneticisini `PSMODE=ENABLED`olarak ayarlayın.

Yayınla

URI ' ye sahip bir HTTP **POST** isteği gönderin:

```
http://hostname:port/context_root/msg/topic/topicString
```

İleti içeriği, *topicString*konu dizgisi kullanılarak yayınlandı.

Abone Ol

URI ' ye sahip bir HTTP **DELETE** isteği gönderin:

```
http://hostname:port/context_root/msg/topic/topicString
```

HTTP içinWebSphere MQ köprüsü, *topicString*konu dizgisine yönelik, yönetilen bir yönetilen olmayan abonelik yaratır. Bir yayınlama döndüğü anda abonelik silinir ya da özel varlık üstbilgisi `x-msg-wait`tarafından ayarlanan bekleme aralığına gelinceye kadar süre bitimi uygulanır.

HTTP örnekleri için WebSphere MQ köprüsünü çalıştırma

HTTP örnekleri için WebSphere MQ köprüsü, yalnızca Windows işletim sisteminde kullanılabilir. Örnekler, HTTP **POST** ve HTTP **DELETE** komutlarının Java programlarından HTTP için WebSphere MQ köprüsünün nasıl gönderileceğini göstermektedir.

Başlamadan önce

Verify your WebSphere MQ bridge for HTTP installation by running step “7” sayfa 986 in “HTTP için WebSphere MQ köprüsünü kurma, yapılandırma ve doğrulama” sayfa 985.

HTTP örnekleri, Çizelge 156 sayfa 990içinde gösterilen dizinlere kurulur. Her durumda, kaynak kod `/src` alt dizinine kurulur.

Çizelge 156. HTTP örneklerinin yeri

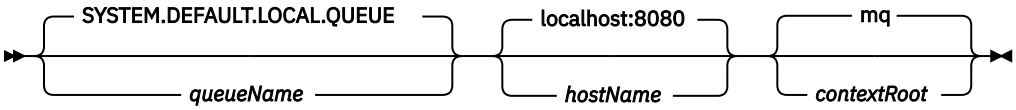
Altyapı	**Konum**
Pencereler	*MQ_INSTALLATION_PATH*/tools/http/samples
Diğer tüm platformlar	*MQ_INSTALLATION_PATH*/samp/http
MQ_INSTALLATION_PATH , WebSphere MQ ' un kurulu olduğu dizini temsil eder.	

Bu görev hakkında

Örnekler, WebSphere MQ AMQSPUT ve AMQSGET örnek uygulamalarının benzetimini sağlar. Bunlar, noktadan noktaya ileti sistemi ortamında aşağıdaki işlevleri gösterirler:

- **HTTPPOST** -HTTP için WebSphere MQ köprüsünü kullanarak ve yanıtları işleten bir WebSphere MQ kuyruğuna ileti koymak için bir Java uygulamasına HTTP **POST** istekleri gönderir.
- **HTTPDELETE** - Sends HTTP **DELETE** requests in a Java application to get messages from a WebSphere MQ queue, using the WebSphere MQ bridge for HTTP and handles the responses containing the WebSphere MQ message.

HTTPPOST ve HTTPDELETEiçin parametreler



HTTPPOST örneğini çalıştırmak için aşağıdaki adımları tamamlayın:

Yordam

1. Bir komut penceresinde, HTTP Samples dizinine gidin.
2. **HTTPPOST** örneğini çalıştırın.

   ```
   java -classpath . HTTPPOST [parameters]
   ```

   HTTPPOST örneği başlatıldığında, aşağıdaki çıkış görüntülenir:

   ```
   HTTP POST Sample start
   Target server is 'hostName'
   Target queue is 'queueName'
   Target context-root is 'contextRoot'
   ```

3. Komut isteminde, ileti gövünüzü oluşturmak istediğiniz metni yazın.
4. İletiyi WebSphere MQ kuyruğuna göndermek için Enter tuşuna basın.
   a) Başka bir ileti göndermek isterseniz, daha fazla metin girin.
   Metinler, ikinci bir WebSphere MQ iletisinin gövdelerini oluşturur.

b) İletiyi WebSphere MQ kuyruğuna göndermek için Enter tuşuna basın.

5. **HTTPPOST**sonuna iki kez basın.

   Aşağıdaki çıkış görüntülenir:

```
HTTP POST Sample end
```

Sonraki adım

HTTPDELETE örneği, WebSphere MQ kuyruğuna yerleştirdiğiniz tüm iletileri yıkıcı bir şekilde alır.

Aşağıdaki adımları tamamlayarak **HTTPDELETE** örneğini çalıştırın:

1. Bir komut penceresinde, *MQ_INSTALLATION_PATH*/tools/samples' a gidin.
   MQ_INSTALLATION_PATH WebSphere MQ ' un kurulu olduğu dizini temsil eder.
2. **HTTPDELETE** örneğini çalıştırın.

```
java -classpath . HTTPPOST [parameters]
```

   HTTPDELETE örneği başlatıldığında, aşağıdaki çıkış görüntülenir:

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

HTTP için WebSphere köprüsü için güvenlik konuları

Bir Web tarayıcısı istemcisinin kimlik doğrulaması için standart Web güvenliği konuları geçerlidir. Authorization to WebSphere MQ resources is at the level of the user running the WebSphere Bridge for HTTP servlet, and not the individual Web browser client. Standart WebSphere MQ güvenliği, WebSphere MQiçin geçerlidir.

HTTP için WebSphere bridge olanağını kullanarak bir Web tarayıcısından bir WebSphere MQ uygulamasına veri akışı ve geri dönüş, üç adımı atmaktadır:

İstemci bağlantısı
HTTP ile bir TCP/IP bağlantısı üzerinden HTTP için WebSphere Bridge 'e tarayıcıdan.

WebSphere MQile kaynak bağdaştırıcısı bağlantısı
The connection is from the WebSphere Bridge for HTTP to a WebSphere MQ queue manager. Bağlantı, bir istemci bağlantısı, TCP/IP üzerinden ya da yerel bir WebSphere MQ bağ tanımları bağlantısı. Bağlantı yapıldıktan sonra, HTTP isteği standart bir yerel kuyruğa ya da iletim kuyruğuna yerleştirilir.

WebSphere MQ yerel kuyruğundan, bir ya da daha çok kanal üzerinden hedef kuyruğa.
Kuyrukları, konuları, kuyruk yöneticilerini ve kanalları güvenli kılmak için standart teknikleri uygulayın.

Yanıt, adımları ters olarak alır.

İstemci bağlantısı

Web taşıyıcısını kullanarak HTTP istemcileri ile uygulama sunucusu arasında güvenli bağlantılar. HTTPS kullanmak gibi standart HTTP sunucu tekniklerini kullanın. Bilgi edinmek için uygulama sunucunuza ilişkin belgelere bakın.

WebSphere MQile kaynak bağdaştırıcısı bağlantısı

Kaynak bağdaştırıcısı ve kuyruk yöneticisi arasındaki bağlantı, yalnızca tek bir kullanıcı kimliği kullanılarak yetkilendirilir. HTTP için WebSphere Köprüsünden gelen istekleri tanımlamak için tek bir kullanıcı kimliği atayın. Kullanıcı kimliğinin, yalnızca dış kullanıcıların erişimleri olması gereken kaynak WebSphere MQ

yetkilerinin olması gerekir. Web güvenliği için standart teknikler kullanarak, gerçek istemciyi ayrı olarak doğrulamanız ve istemciyle art arda etkileşimler için güven oluşturmanız gerekir.

Tekli kullanıcı kimliğini kullanarak, kaynak bağdaştırıcısı ile kuyruk yöneticisi arasındaki bağlantıyı güvenli hale getirin. Kullanıcı kimliğinin, kuyruklara ve konulara ilişkin iletileri okumak ve yazmak için gerekenden daha fazla bilgi sahibi olmamasını engelle. HTTP için WebSphere Bridge, İnternet ile intranetiniz arasındaki bir saldırı noktasıdır.

Kaynak bağdaştırıcınız ile WebSphere MQ arasındaki bağlantıyı nasıl güvenceye aldıysanız, belirli kaynak bağdaştırıcınızla birlikte değişir. Refer to the documentation for the resource adapter.

Component Object Model Interface olanağının kullanılması (ActiveXiçin WebSphere MQ Automation Sınıfları)

WebSphere MQ Automation Classes for ActiveX (MQAX), uygulamanıza WebSphere MQ' a erişmek için kullanabileceğiniz sınıfları sağlayan ActiveX bileşenleridir.

MQAX, iletişim kurmak için bir WebSphere MQ ortamı ve buna karşılık gelen bir WebSphere MQ uygulaması gerektirir.

Bu, ActiveX uygulamanızı WebSphere MQaracılığıyla erişebileceğiniz kurumsal sistemlerinizin herhangi birinde işlem çalıştırma ve verilere erişme yeteneği verir.

ActiveXiçinWebSphere MQ Otomasyon Sınıfları:

- WebSphere MQ API ' nın işlevlerine ve özelliklerine erişmenizi sağlar; diğer WebSphere MQ altyapılarına tam bağlanırlık sağlar.
- Bir ActiveX bileşeni için beklenen olağan sözleşmeler ile uyumludur.
- .NET, C + +, Java ve LotusScriptiçin de kullanılabilir olan WebSphere MQ nesne modeline uygun bir şekilde uyumludur.

MQAX başlangıç örnekleri sağlanır. Bu örnekleri başlangıçta, MQAX kurulumunuzun başarılı olup olmadığını ve temel WebSphere MQ ortamını bulunca yerine getirdiğinizi denetlemek için kullanabilirsiniz. Ayrıca, MQAX 'in nasıl kullanılabileceği de gösterir.

COM ve ActiveX komut dosyası oluşturma

Component Object Model (COM), Microsofttarafından tanımlanan, nesne tabanlı bir programlama modelidir. Yazılım bileşenlerinin, yazıldığı ya da konumlarının bulunduğu bilgisayar dilinden bağımsız olarak birbirlerinin yerini belirleyebilmelerini ve birbirleriyle iletişim kurmalarını sağlayan bir şekilde nasıl sağlanabileceğini belirtir.

ActiveX is a set of technologies, based on COM, that integrates application development, reusable components, and Internet technologies on the Microsoft Pencereler platforms. ActiveX bileşenleri, uygulamalar tarafından dinamik olarak erişilebilecek arabirimler sağlar. ActiveX komut dosyası istemcisi, örneğin, ActiveX (ya da COM) bileşenleri tarafından sağlanan arabirimleri kullanan bir program ya da komut dosyası oluşturabilen ya da yürütebilen bir uygulamadır.

WebSphere MQ ortamı desteği

WebSphere MQ Automation Classes for ActiveX can only be used with **32 bit** ActiveX scripting clients.

COM bileşeni yalnızca **32 bit** uygulamalar için kullanılabilir. 64 bit COM uygulaması yazmak istiyorsanız, .NET arabirimini kullanabilirsiniz.

MQAX 'i bir WebSphere MQ sunucusu ortamında çalıştırmak için sisteminizde Windows 2000 ya da üstü kurulu olmalıdır.

To run the MQAX in a WebSphere MQ MQI client environment you need WebSphere MQ MQI client on Pencereler 2000 or later installed on your system:

WebSphere MQ MQI istemcisi en az bir WebSphere MQ Server sunucusuna erişimi gerektirir. When both the WebSphere MQ MQI client and the WebSphere MQ server are installed on your system, MQAX applications always run against the server. MQAI ' ye ActiveX arabirimi yalnızca WebSphere MQ sunucusu ortamlarında kullanılabilir.

ActiveXiçin WebSphere MQ Otomasyon Sınıflarını kullanarak tasarlama ve programlama

ActiveX dışı uygulamalara erişen MQAX uygulamalarının tasarlanması

WebSphere MQ Otomasyon Sınıfları, WebSphere MQ API 'sının işlevlerine erişim sağlar. Bu nedenle, WebSphere MQ olanağını kullanarak Windows uygulamanızı getirebilecek tüm avantajlardan yararlanabilirsiniz.

Uygulamanızın genel tasarımı, herhangi bir WebSphere MQ uygulaması için aynıdır, bu nedenle, “Uygulamaların geliştirilmesi” sayfa 7 bölümünde açıklanan tüm tasarım özelliklerini göz önünde bulundurun.

WebSphere MQ Otomasyon Sınıflarını kullanmak için, uygulamanızın içindeki Windows programlarını, COM nesnelerinin yaratılmasını ve kullanılmasını destekleyen bir dil kullanarak kodlatın. Örneğin, Visual Basic, Java ve diğer ActiveX komut dosyası istemcileri. Daha sonra, gereksinim duyardığınız WebSphere MQ nesneleri uygulama dilinin yerli sözdizimiyle kodlanabileceğinden, sınıfların uygulamanıza kolayca entegre edilebilmesini sağlar.

ActiveXiçin WebSphere MQ Otomasyon Sınıflarını Kullanma

ActiveXiçin WebSphere MQ Otomasyon Sınıflarını kullanan bir ActiveX uygulaması tasarlarken, bilgilerin en önemli öğesi, uzak WebSphere MQ sisteminden gönderilen ya da alınan iletidir. Bu nedenle, iletiye eklenen öğelerin biçimini bilmelisiniz. Bir MQAX komut dosyası için bir iş için, iletiyi alan ya da gönderen WebSphere MQ uygulaması, ileti yapısını bilmelidir.

MQAX uygulamasıyla bir ileti gönderiyorsanız ve MQAX uçunda veri dönüştürme işlemi gerçekleştirmek istiyorsanız, şunları bilmeniz gerekir:

- Uzak sistem tarafından kullanılan kod sayfası
- Uzak sistem tarafından kullanılan kodlama

kodunuzu taşınabilir tutmak için, kod sayfası ve kodlamayı ayarlamak iyi bir uygulamadır. şu anda hem gönderme hem de alma sistemlerinde aynı anda aynı olsa bile.

Tasarladığınız sistemin uygulamasını nasıl yapılandıracağını göz önünde bulundurarak, MQAX komut dosyalarınızın WebSphere MQ kuyruk yöneticisi ya da WebSphere MQ istemcisi kurulu olduğu makineden farklı olarak çalıştırıldığını unutmayın.

Programlama ipuçları ve öneriler

Aşağıdaki ipuçları ve ipuçları önemli bir sırada yer almaz. Bunlar, yaptığınız işe uygun bir şekilde, size zaman kazanmanızı sağlar.

İleti Tanımlayıcısı özellikleri

Bir programdaki ileti tanımlayıcı özelliklerini değiştirdiğinizde, alanların onaltılı eşdeğerlerini kullanmanız daha iyi olabilir.

Bu bölümdeki bilgiler aşağıdaki özelliklere başvurur:

- AccountingToken
- CorrelationId
- GroupId
- MessageId

Where a WebSphere MQ application is the originator of a message and WebSphere MQ generates these properties, it is better to use the AccountingTokenHex, CorrelationIdHex, GroupIdHex, and MessageIdHex properties if you want to look at their values, or manipulate them in any way, including passing them back in a message to WebSphere MQ. Bunun nedeni, WebSphere MQ tarafından oluşturulan değerlerin, 0 ile 255 arasında herhangi bir değere sahip olan bayt dizgileri olduğunu, bunlar da yazdırılabilir karakterlerin dizgileri olmalarıdır.

MQAX komut dosyanızın bir iletinin kaynağı olduğu durumlarda, AccountingToken, CorrelationId, GroupIdve MessageId özelliklerini ya da Hex eşdeğerlerini kullanabilirsiniz.

WebSphere MQ değişmezleri

WebSphere MQ constants are provided as members of the enum WebSphere MQ in library MQAX200.

WebSphere MQ dizgi değişmezleri

WebSphere MQ dizgi değişmezleri ve bunlara karşılık gelen karakter dizilimleri.

WebSphere MQ string constants are not available when using WebSphere MQ Automation Classes for ActiveX. Aşağıdaki listede gösterilen açık karakter dizesini ve ihtiyacınız olabilecek diğer kişileri kullanmanız gerekir. Komutların boşluk kullanılarak sekiz karaktere kadar doldurulması gerekir:

MQFMT_NONE	" "
MQFMT_ADMIN	"MQADMIN"
MQFMT_CHANNEL_COMPLETED	"MQCHCOM"
MQFMT_CICS	"MQCICS"
MQFMT_COMMAND_1	"MQCMD1 "
MQFMT_COMMAND_2	"MQCMD2 "
MQFMT_DEAD_LETTER_HEADER	"MQDEAD"
MQFMT_DIST_HEADER	"MQHDIST"
MQFMT_OLAY	"MQOLAY"
MQFMT_IMS	"MQIMS"
MQFMT_IMS_VAR_STRIsNG	"MQIMSVS"
MQFMT_MD_EXTENSION	"MQHMDE"
MQFMT_PCF	"MQPCF"
MQFMT_REF_MSG_HEADER	"MQHREF"
MQFMT_RF_HEADER	"MQHRF"
MQFMT_STRING	"MQSTR"
MQFMT_TETIKLEYICISI	"MQTRIG"
MQFMT_WORK_INFO_HEADER	"MQHIH"
MQFMT_XMIT_Q_HEADER	"MQXMIT"

Boş dizgi değişmezleri

Dört MQMessage özelliklerinin kullanıma hazırlanması için kullanılan WebSphere MQ değişmezleri, MQMI_NONE (24 NULL karakter), MQCI_NONE (24 NULL karakter), MQGI_NONE (24 NULL karakter) ve MQACT_NONE (32 NULL karakter), ActiveXiçin WebSphere MQ Otomasyon Sınıfları tarafından desteklenmez. Bunları boş dizgilerle ayarlamak aynı etkiye sahiptir.

Örneğin, bir MQMessage 'ın çeşitli tanıtıcılarını şu değerlere ayarlayın: *mymessage*.**MessageId** = "" *mymessage*.**CorrelationId** = "" *mymessage*.**AccountingToken** = ""

WebSphere MQ' dan ileti alma

WebSphere MQ' dan bir ileti almanın çeşitli yolları vardır:

- Bir GET işlemi, Visual Basic TIMER işlevini kullanarak, bir GET işlemi yayınlayarak yoklanıyor.
- Issuing a GET with the Wait option; you specify the wait duration by setting the WaitInterval property. Sisteminizi çok iş parçacıklı bir ortamda çalışacak şekilde ayarlamanıza rağmen, o sırada çalışan yazılımların yalnızca tek iş parçacıklı çalışabileceğini göz önünde bulundurun. Bu, süresiz olarak sistem kilitlenmesini önler.

  Diğer iş parçacıkları etkilenmeden çalışır. Ancak, diğer iş parçacıklarınız WebSphere MQ' ya erişim gerektiriyorsa, ek MQAX kuyruk yöneticisi ve kuyruk nesneleri kullanarak WebSphere MQ için ikinci bir bağlantı gerektirir.

Bekleme seçeneğiyle GET komutu verilmesi ve WaitInterval ile MQWI_UNESSıNIN değerine ayarlanması, işlemin tek iş parçacıklı olması durumunda GET çağrısına kadar sistemin kilitlenmesine neden olur.

Veri dönüştürmenin kullanılması

Two forms of data conversion are supported by WebSphere MQ Automation Classes for ActiveX - numeric encoding, and character set conversion.

Sayısal kodlama

MQMessage Encoding özelliğini ayarlarsanız, farklı sayısal kodlama sistemleri arasında aşağıdaki yöntemler dönüştürülür:

- ReadDecimal2 yöntemi
- ReadDecimal4 yöntemi
- ReadDouble yöntemi
- ReadDouble4 yöntemi
- ReadFloat yöntemi
- ReadInt2 yöntemi
- ReadInt4 yöntemi
- ReadLong yöntemi
- ReadShort yöntemi
- ReadUInt2 yöntemi
- WriteDecimal2 yöntemi
- WriteDecimal4 yöntemi
- WriteDouble yöntemi
- WriteDouble4 yöntemi
- WriteFloat yöntemi
- WriteInt2 yöntemi
- WriteInt4 yöntemi
- WriteLong yöntemi
- WriteShort yöntemi
- WriteUInt2 yöntemi

Kodlama özelliği, sağlanan WebSphere MQ değişmezleri kullanılarak ayarlanabilir ve yorumlanabilir. Şekil 216 sayfa 996 şunlara bir örnek gösterir:

```
/* Encodings for Binary Integers */
        MQENC_INTEGER_UNDEFINED
        MQENC_INTEGER_NORMAL
        MQENC_INTEGER_REVERSED

/* Encodings for Decimals */
        MQENC_DECIMAL_UNDEFINED
        MQENC_DECIMAL_NORMAL
        MQENC_DECIMAL_REVERSED

/* Encodings for Floating-Point Numbers */
        MQENC_FLOAT_UNDEFINED
        MQENC_FLOAT_IEEE_NORMAL
        MQENC_FLOAT_IEEE_REVERSED
        MQENC_FLOAT_S390
```

Şekil 216. Kodlama için sağlanan WebSphere MQ değişmezleri

For example, to send an integer from an Intel system to a System/390 operating system in System/390 encoding:

```
Dim msg As New MQMessage   'Define a WebSphere MQ message for our use..
Print msg.Encoding    'Currently 546 (or X'222')
                                          'Set the encoding property
                                          to 785 (or X'311')
msg.Encoding = MQENC_INTEGER_NORMAL OR MQENC_DECIMAL_NORMAL
                        OR MQENC_FLOAT_S390
Print msg.Encoding    'Print it to see the change
Dim local_num As long 'Define a long integer
local_num = 1234                          'Set it
msg.WriteLong(local_num)         'Write the number into the message
```

Karakter kümesi dönüştürmesi

Bir sistemden başka bir sisteme ileti gönderdiğinizde, kod sayfalarının farklı olduğu bir ileti gönderdiğinizde karakter takımı dönüştürme gereklidir. Kod sayfası dönüşümü aşağıdaki gibi kullanılır:

- ReadString yöntemi
- ReadNullTerminatedString yöntemi
- WriteString yöntemi
- WriteNullTerminatedString yöntemi
- MessageData özelliği

MQMessage CharacterSet özelliğini desteklenen bir karakter kümesi değerine (CCSID) ayarlamalısınız.

ActiveX için WebSphere MQ Otomasyon Sınıfları, karakter kümesi dönüştürmesi gerçekleştirmek için dönüştürme tablolarını kullanır.

Örneğin, dizgileri otomatik olarak kod sayfası 437 'ye dönüştürmek için:

```
Dim msg As New MQMessage                'Define a WebSphere MQ message
msg.CharacterSet = 437                'Set code page required
msg.WriteString "A character string"'Put character string in message
```

WriteString yöntemi, dizgi verilerini (örnekte "A karakter dizgisi") bir Unicode dizgi olarak alır. Daha sonra, 34B001B5.TBLdönüştürme çizelgesini kullanarak bu verileri Unicode 'dan kod sayfası 437 'ye dönüştürür.

Unicode diziliminde, kod sayfası 437 tarafından desteklenmeyen karakterler, kod sayfası 437 'den standart yerine koyma karakteri olarak verilir.

Benzer şekilde, ReadString yöntemini kullandığınızda, gelen iletinin WebSphere MQ Message Descriptor (MQMD) değeri tarafından kurulmuş bir karakter kümesi vardır ve komut dosyası dilinize geçirilmeden önce bu kod sayfasından Unicode 'a dönüştürme vardır.

Threading

ActiveX için WebSphere MQ Automation Sınıfları, nesnelerin iş parçacıkları arasında kullanılabileceği bir serbest threading modelini uygular.

MQAX, MQQueue ve MQQueueManager nesnelerinin kullanılmasına izin verirken, WebSphere MQ şu anda farklı iş parçacıkları arasında çekme noktalarının paylaşılmasına izin vermez.

Başka bir iş parçacığı üzerinde bunları kullanma girişimleri hatayla sonuçlanabilir ve WebSphere MQ , MQRC_HCONN_ERROR dönüş kodunu döndürür.

Not: İşlem başına yalnızca bir MQSession nesnesi var. Using the MQSession CompletionCode and ReasonCode is not recommended in multithreaded environments. MQSession hata değerlerinin üzerine, ilk iş parçacığıdaki yükseltilmiş ve denetlenmekte olan bir hata arasındaki ikinci bir iş parçacığı tarafından yazılabilir. İş parçacıkları, her bir yöntem çağrısı ya da özellik erişimi süresi için diziselleştirilir. Bu nedenle, Bekleme seçeneği ile Alma seçeneği verilmesi, MQAX nesnelerine erişen diğer iş parçacıklarının işlem tamamlanıncaya kadar askıya alınmasına neden olur.

Hata işleme

Bu bilgilerde, MQAX nesnesi özellikleri, hata işleme yöntemi, kural dışı durumların nasıl yükseltildiğini açıklayan kurallar ve bir özellik alınması açıklanır.

Her MQAX nesnesi, hata bilgilerinin tutulacağı özellikleri ve ilk duruma getirme ya da temizleme yöntemini içerir. Özellikler şunlardır:

- CompletionCode
- ReasonCode
- ReasonName

Yöntem şöyledir:

- ClearErrorKodları

Hata işleme yöntemi

MQAX komut dosyası ya da uygulamanız bir MQAX nesnesinin yöntemini çağırır ya da MQAX nesnesinin bir özelliğine erişir ya da bu nesneye erişir:

1. İlgili nesnelerde ReasonCode ve CompletionCode değeri güncellenir.
2. MQSession nesnesindeki ReasonCode ve CompletionCode , aynı bilgilerle güncellenir.

   Not: İş parçacıklı uygulamalarda MQSession hata kodlarının kullanımına ilişkin kısıtlamalar için bkz. “Threading” sayfa 997 .

   CompletionCode , MQSession 'ın ExceptionThreshold özelinden büyük ya da ona eşitse, MQAX kural dışı durum yayınlar (sayı 32000). Bunu işlemek için On Error (ya da eşdeğer) deyimini kullanarak komut dosyanızın içinde kullanın.
3. Bu forma sahip olan ilişkili hata dizgisini almak için Error işlevini kullanın:

   ```
   MQAX: CompletionCode=xxx, ReasonCode=xxx, ReasonName=xxx
   ```

   On Error deyimlerini kullanmaya ilişkin daha fazla bilgi için, ActiveX komut dosyası dilinize ilişkin belgelere bakın.

   MQSession nesnesindeki CompletionCode ve ReasonCode kullanılarak, basit hata işleyiciler için uygun bir değer vardır.

   ReasonName özelliği, ReasonCode' un geçerli değeri için WebSphere MQ simgesel adını döndürür.

Özel durumlar oluştu

Aşağıdaki kurallar, kural dışı durumların nasıl işlendiğini açıklar:

- Bir özellik ya da yöntem, tamamlanma kodunu, kural dışı durum eşiğinden büyük ya da ona eşit bir değere ayarladığında (genellikle 2 olarak ayarlanır) bir kural dışı durum oluşturulur.
- Tüm yöntem çağrıları ve özellik kümeleri tamamlama kodunu ayarlar.

Özellik alma

CompletionCode ve ReasonCode her zaman güncellenmediği için bu, özel bir olaydır:

- Bir özellik başarılı olursa, nesne ve MQSession nesnesi ReasonCode ve CompletionCode değiştirilmeden kalır.
- If a property get fails with a CompletionCode of warning, the ReasonCode and CompletionCode remain unchanged.
- Bir özellik CompletionCode hata ile başarısız olursa, ReasonCode ve CompletionCode , doğru değerleri yansıtacak şekilde güncellenir ve hata işleme devam edilir olarak işlenir.

MQSession sınıfının bir WebSphere MQ neden kodunu simgesel bir adla değiştirmek için kullanılabilecek bir yöntemi (*ReasonCodeName*) var. Bu, özellikle beklenmedik hataların ortaya çıkabileceği programlar geliştirirken kullanışlıdır. Ancak, ad, kullanıcılara sunum yapmak için ideal değildir.

Her sınıfın, o sınıfa ilişkin geçerli neden kodunun simgesel adını döndüren *ReasonName*özelliği de vardır.

ActiveX başvurusu için WebSphere MQ Otomasyon Sınıfları

This section describes the classes of the WebSphere MQ Automation Classes for ActiveX (MQAX), developed for ActiveX. The classes enable you to write ActiveX applications that can access other applications running in your non-ActiveX environments, using WebSphere MQ.

WebSphere MQ Automation Classes for ActiveX interface

WebSphere MQ Automation Classes for ActiveX provides predefined numeric ActiveX constants (such as MQMT_REQUEST) needed to use the classes.

ActiveX otomasyon sınıfları aşağıdaki gibi oluşur:

- “MQSession Sınıfı” sayfa 1000
- “MQQueueManager sınıfı” sayfa 1002
- “MQQueue sınıfı” sayfa 1013
- “MQMessage sınıfı” sayfa 1028
- “MQPutMessageSeçenekleri sınıfı” sayfa 1049
- “MQGetMessageSeçenekleri sınıfı” sayfa 1051
- “MQDistributionList sınıfı” sayfa 1053
- “MQDistributionListÖğe sınıfı” sayfa 1057

In addition, WebSphere MQ Automation Classes for ActiveX provides predefined numeric ActiveX constants (such as MQMT_REQUEST) needed to use the classes. Bunlar, MQAX200kitaplığındaki sıralı değer listesi MQ ' da sağlanır. The constants are a subset of those defined in the WebSphere MQ C header files (cmqc*.h) with some additional WebSphere MQ Automation Classes for ActiveX Reason codes.

ActiveX sınıfları için WebSphere MQ Otomasyon Sınıfları Hakkında

Bu bilgileri, Developing applications reference(Uygulamalar başvurusu) altındaki başvuru konularının yanında okuyun.

Önemli bilgi için Yalnızca Windows 'ta birincil kuruluşla kullanılabilen özellikler konusuna bakın.

MQSession sınıfı, MQAX nesnelerinden herhangi birinde gerçekleştirilen son işlemin durumunu içeren bir kök nesne sağlar. Daha fazla bilgi için bkz. “Hata işleme” sayfa 997 .

MQQueueManager ve MQQueue sınıfları, temeldeki WebSphere MQ nesnelerine erişim sağlar. Methods or property accesses for these classes in general result in calls being made across the WebSphere MQ MQI.

MQMessage, MQPutMessageSeçenekleri ve MQGetMessageSeçenekleri sınıfları, MQMD, MQPMO ve MQGMO veri yapılarını sarmala ve kuyruklara ileti göndermenize ve bunlara ileti göndermenize yardımcı olmak için kullanılır.

MQDistributionList sınıfı, çıkışa ilişkin bir kuyruklar derlemini (yerel, uzak ya da diğer ad) sarsalıyor. MQDistributionListÖğe sınıfı, MQOR, MQRR ve MQPMR yapılarını sarsalıyor ve bunları sahip olan bir dağıtım listesiyle ilişkilendirir.

Parametre geçişi

Yöntem çağırımlarındaki parametreler, parametrenin bir nesne olduğu durumlar dışında, geçirilen bir başvurudur. Bu durumda, geçirilen bir nesne, bu parametrenin bir nesneden olduğu durumlar dışında,

Belirtilen sınıf tanımlamaları, her değiştirge ya da özellik için Veri Tipi listesini içerir. Visual Basic gibi birçok ActiveX istemcisi için, kullanılan değişken gerekli tipte değilse, değer otomatik olarak gerekli tipten ya da bu tür bir dönüştürmeye dönüştürülmüş olur. İstemcinin standart kuralları şöyledir; MQAX bu tür bir dönüştürmeyi sağlar.

Yöntemlerin çoğu değişmez uzunluklu dizgi değiştirgeleri alır ya da değişmez uzunluklu bir karakter dizilimi döndürür. Dönüştürme kuralları şunlardır:

- Kullanıcı, bir giriş parametresi olarak ya da bir dönüş değeri olarak yanlış uzunluğun değişmez uzunluklu bir dizgisi sağladıysa, değer kısaltılır ya da gerektiği şekilde sondaki boşluklarla doldurulr.
- Kullanıcı, giriş parametresi olarak yanlış uzunluğun değişken uzunluklu bir dizilimini sarf ettiyse, değer kesilerek ya da sondaki boşluklarla doldurulsa doldurulur.
- Kullanıcı, dönüş değeri olarak yanlış uzunluğa ait bir değişken uzunluklu bir dizgi varsa, dizgi gereken uzunluğa göre ayarlanır (çünkü bir değer döndürülürse, dizedeki önceki değeri yok eder).
- Giriş değiştirgeleri olarak sağlanan dizgiler gömülü boş değer içerebilir.

Bu sınıflar, MQAX200 kitaplığında bulunabilir.

Nesne erişim yöntemleri

Bu yöntemler, doğrudan herhangi bir WebSphere MQ çağrısını ilişkilendirmez. Each of these methods creates an object in which reference information is then held, followed by connecting to or opening a WebSphere MQ object:

Bir kuyruk yöneticisine bağlantı yapıldığında, WebSphere MQtarafından oluşturulan 'connection handle' özniteliğini tutar.

Bir kuyruk açıldığında, WebSphere MQtarafından oluşturulan 'object handle' özniteliğini içerir.

Bu WebSphere MQ öznitelikleri, MQAX programının doğrudan kullanımına sunulmaz.

Hatalar

Parametre geçişindeki sözdizimsel hatalar, ActiveX istemcisi tarafından derleme sırasında ve çalıştırma sırasında saptlanabilir. Hatalar Visual Basic 'te Error (Hata) ile kapana kısılabilir.

WebSphere MQ ActiveX sınıfları, yalnızca iki özel salt okunur özelliği (ReasonCode ve CompletionCodeiçerir) içerir. Bu özellikler her zaman okunabilir.

Diğer herhangi bir özelliğe erişme ya da herhangi bir yöntem çağrısını yayınlamaya ilişkin bir girişim WebSphere MQ' dan bir hata oluşturabilir.

Bir özellik kümesi ya da yöntem çağrısı başarılı olursa, sahip nesnenin ReasonCode değeri MQRC_NONE olarak ayarlanır ve CompletionCode MQCC_OK olarak ayarlanır.

Özellik erişimi ya da yöntem çağrısı başarılı olamazsa, neden ve tamamlanma kodları bu alanlarda ayarlanır.

MQSession Sınıfı

Bu, ActiveXiçin WebSphere MQ Automation Sınıfları için kök sınıftır.

Her zaman ActiveX istemci işlemi başına yalnızca bir MQSession nesnesi vardır. İkinci bir nesne yaratma girişimi, özgün nesneye ilişkin ikinci bir başvuru yaratır.

Yaratma

Yeni seçeneği, yeni bir MQSession nesnesi yaratır.

Sözdizimi

dim *mqtakıntı* **Yeni Olarak MQSession Küme** *mqtaks* = **Yeni MQSession**

Özellikler

- “CompletionCode özelliği” sayfa 1000.
- “ExceptionThreshold özelliği” sayfa 1001.
- “ReasonCode özelliği” sayfa 1001.
- “ReasonName özelliği” sayfa 1001.

Yöntem

- “AccessGetMessageOptions yöntemi” sayfa 1001.
- “AccessMessage yöntemi” sayfa 1002.
- “AccessPutMessageOptions yöntemi” sayfa 1002.
- “AccessQueueManager yöntemi” sayfa 1002.
- “ClearErrorCodes yöntemi” sayfa 1002.
- “ReasonCodeAd yöntemi” sayfa 1002.

CompletionCode özelliği

Salt okunur. Herhangi bir WebSphere MQ nesnesiyle ilgili olarak yayınlanan en son yöntem ya da özellik kümesi tarafından ayarlanan WebSphere MQ tamamlanma kodunu döndürür.

Bir yöntem ya da özellik kümesi herhangi bir MQAX nesnesine başarıyla çağrıldığında MQCC_OK değerine sıfırlanır.

Bir hata olayı işleyicisi, hangi nesnenin dahil olduğunu bilmek zorunda kalmadan hatayı tanılamak için bu özelliği inceleyebilir.

MQSession nesnesindeki CompletionCode ve ReasonCode kullanılarak, basit hata işleyiciler için çok uygun bir durum vardır.

Not: İş parçacıklı uygulamalarda MQSession hata kodlarının kullanımına ilişkin kısıtlamalar için “Threading” sayfa 997 ' e bakın.

Tanımlı: MQSession sınıfı

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi:

Almak için: *completioncode & = MQSession*.CompletionCode

ExceptionThreshold özelliği

Okuma-yazma. MQAX 'in bir kural dışı durum yayınlayacağı WebSphere MQ hatasının düzeyini tanımlar. Varsayılan olarak MQCC_FAILED değerine ayarlanır. A value greater than MQCC_FAILED effectively prevents exception processing, leaving the programmer to perform checks on the CompletionCode and ReasonCode.

Tanımlı: MQSession sınıfı

Veri Tipi: Uzun

Değerler:

- Herhangi biri, ancak MQCC_UYARI, MQCC_FAILED ya da daha büyük bir değer düşünün.

Sözdizimi:

Almak için: *ExceptionThreshold& = MQSession*.**ExceptionThreshold**

Ayarlamak için: *MQSession.***ExceptionThreshold** = *ExceptionThreshold$*

ReasonCode özelliği

Salt okunur. Herhangi bir WebSphere MQ nesnesi için yayınlanan en son yöntem ya da özellik kümesiyle ayarlanan neden kodunu döndürür.

Bir hata olayı işleyicisi, hangi nesnenin dahil olduğunu bilmek zorunda kalmadan hatayı tanılamak için bu özelliği inceleyebilir.

MQSession nesnesindeki CompletionCode ve ReasonCode kullanılarak, basit hata işleyiciler için çok uygun bir durum vardır.

Not: İş parçacıklı uygulamalarda MQSession hata kodlarının kullanımına ilişkin kısıtlamalar için “Threading” sayfa 997 ' e bakın.

Tanımlı: MQSession sınıfı

Veri Tipi: Uzun

Değerler:

- “Neden kodları” sayfa 1064altında listelenen ek MQAX değerleri için bkz. Neden (MQlong) ve ek MQAX değerleri.

Sözdizimi: almak için: *reasoncode & = MQSession*.**ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Not: İş parçacıklı uygulamalarda MQSession hata kodlarının kullanımına ilişkin kısıtlamalar için “Threading” sayfa 997 ' e bakın.

Tanımlı: MQSession sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQSession*.**ReasonName**

AccessGetMessageOptions yöntemi

Yeni bir MQGetMessageSeçenekleri nesnesi yaratır.

Tanımlı: MQSession sınıfı

Sözdizimi: *gmo = MQSession*.**AccessGetMessageOptions()**

AccessMessage yöntemi

Yeni bir MQMessage nesnesi yaratır.

Tanımlı: MQSession sınıfı

Sözdizimi: *msg = MQSession*.**AccessMessage()**

AccessPutMessageOptions yöntemi

Yeni bir MQPutMessageSeçenekleri nesnesi yaratır.

Tanımlı: MQSession sınıfı

Sözdizimi: *pmo = MQSession*.**AccessPutMessageOptions()**

AccessQueueManager yöntemi

Yeni bir MQQueueManager nesnesi yaratır ve bu nesneyi, WebSphere MQ MQI istemcisi ya da WebSphere MQ sunucusu aracılığıyla gerçek bir kuyruk yöneticisine bağlamaktadır. Bir bağlanma işlemi gerçekleştirdikçe, bu yöntem kuyruk yöneticisi nesnesi için de bir açma işlemi gerçekleştirir.

Sisteminizde WebSphere MQ MQI istemcisi ve WebSphere MQ sunucusu kurulu olduğunda, MQAX uygulamaları varsayılan olarak sunucuya karşı çalışır. MQAX 'i istemciye karşı çalıştırmak için, istemci bağ tanımları kitaplığı `GMQ_MQ_LIB` ortam değişkeninde belirtilmeli; örneğin, `GMQ_MQ_LIB=mqic.dll`.

Yalnızca istemci kuruluşu için, `GMQ_MQ_LIB` ortam değişkeninin ayarlanması gerekli değildir. Bu değişken ayarlanmadığında, WebSphere MQ amqzst.dlldosyasını yüklemeyi dener. Bu DLL yoksa (istemcinin yalnızca kuruluşunda olduğu gibi), WebSphere MQ mqic.dlldosyasını yükleme girişiminde bulunur.

Başarılı olursa, MQQueueManager' ın ConnectionStatus değerini TRUE olarak ayarlıyor.

Kuyruk yöneticisi, ActiveX yönetim ortamı başına en çok bir MQQueueManager nesnesine bağlanabilir.

Kuyruk yöneticisine yönelik bağlantı başarısız olursa, bir hata olayı oluşturulur ve MQSession nesnesinin ReasonCode ve CompletionCode ayarı ayarlanır.

Tanımlı: MQSession sınıfı

Sözdizimi: *set qm = MQSession*.**AccessQueueManager** (*Name$*)

Parametre:*Name$* String. Bağlanılacak Kuyruk Yöneticisinin adı.

ClearErrorCodes yöntemi

CompletionCode öğesini MQCC_OK değerine ve ReasonCode ' ı MQRC_NONE olarak sıfırlar.

Tanımlı: MQSession sınıfı

Sözdizimi: Call *MQSession*.**ClearErrorCodes ()**

ReasonCodeAd yöntemi

Belirtilen sayısal değer ile neden kodunun adını döndürür. Kullanıcılara hata durumlarına ilişkin daha net göstergeler vermek yararlı olur. Ad hala bir şekilde şifreli (örneğin, ReasonCodeAd (2059) **MQRC_Q_MGR_NOT_AVAM**); bu nedenle, olası hataların yakalanması ve uygulamaya uygun açıklayıcı metinle değiştirilmesi gerekir.

Tanımlı: MQSession sınıfı

Sözdizimi: *errname $= MQSession*.**ReasonCodeAd**(*reasonCode&*)

Parametre:*reasoncode &* Long. Simgesel adın gerekli olduğu neden kodu.

MQQueueManager sınıfı

Bu sınıf, kuyruk yöneticisine yönelik bir bağlantıyı temsil eder. Kuyruk yöneticisi yerel olarak çalıştırılabilir (bir WebSphere MQ sunucusu) ya da WebSphere MQ istemcisi tarafından sağlanan erişim ile uzaktan çalıştırılabilir. Bir uygulama bu sınıfın bir nesnesini yaratmalı ve bunu bir kuyruk yöneticisine bağlamalıdır. Bu sınıfın bir nesnesi yok edildiğinde, bu sınıfın bir nesnesi otomatik olarak kuyruk yöneticisinden çıkarılır.

Bulundurma

MQQueue sınıfı nesneleri bu sınıfla ilişkilendirilir.

Yeni, yeni bir MQQueueManager nesnesi yaratır ve tüm özellikleri ilk değerlerine ayarlar. Diğer bir seçenek olarak, MQSession sınıfının AccessQueueManager yöntemini kullanabilirsiniz.

Yaratma

Yeni, bir **yeni** MQQueueManager nesnesi yaratır ve tüm özellikleri ilk değerlerine ayarlar. Diğer bir seçenek olarak, MQSession sınıfının AccessQueueManager yöntemini kullanabilirsiniz.

Sözdizimi

Dim *mgr* **Yeni MQQueueManager kümesi** *mgr* = **Yeni MQQueueManager**

Özellikler

- "AlternateUsertanıtıcı özelliği" sayfa 1004.
- "AuthorityEvent özelliği" sayfa 1005.
- "BeginOptions özelliği" sayfa 1005.
- "ChannelAutoTanımlaması özelliği" sayfa 1005.
- "ChannelAutoDefinitionEvent özelliği" sayfa 1005.
- "ChannelAutoDefinitionExit özelliği" sayfa 1006.
- "CharacterSet özelliği" sayfa 1006.
- "CloseOptions özelliği" sayfa 1006.
- "CommandInputQueueName özelliği" sayfa 1006.
- "CommandLevel özelliği" sayfa 1006.
- "CompletionCode özelliği" sayfa 1006.
- "ConnectionHandle özelliği" sayfa 1007.
- "ConnectionStatus özelliği" sayfa 1007.
- "ConnectOptions özelliği" sayfa 1007.
- "DeadLetterQueueName özelliği" sayfa 1007.
- "DefaultTransmissionQueueName özelliği" sayfa 1008.
- "Açıklama özelliği" sayfa 1008.
- "DistributionLists özelliği" sayfa 1008.
- "InhibitEvent özelliği" sayfa 1008.
- "IsConnected özelliği" sayfa 1008.
- "IsOpen özelliği" sayfa 1009.
- "LocalEvent özelliği" sayfa 1009.
- "MaximumHandles özelliği" sayfa 1009.
- "MaximumMessageUzunluk özelliği" sayfa 1009.
- "MaximumPriority özelliği" sayfa 1009.
- "MaximumUncommittedİletileri özelliği" sayfa 1009.
- "Ad özelliği" sayfa 1010.

- “ObjectHandle özelliği” sayfa 1010.
- “PerformanceEvent özelliği” sayfa 1010.
- “Platform özelliği” sayfa 1010.
- “ReasonCode özelliği” sayfa 1010.
- “ReasonName özelliği” sayfa 1011.
- “RemoteEvent özelliği” sayfa 1011.
- “StartStopOlay özelliği” sayfa 1011.
- “SyncPointKullanılabilirlik özelliği” sayfa 1011.
- “TriggerInterval özelliği” sayfa 1011.

Yöntemler

- “AccessQueue yöntemi” sayfa 1012.
- “AddDistributionListe yöntemi” sayfa 1012.
- “Geriletme yöntemi” sayfa 1012.
- “Başlangıç yöntemi” sayfa 1013.
- “ClearErrorCodes yöntemi” sayfa 1013.
- “Kesinleştirme yöntemi” sayfa 1013.
- “Bağlanma yöntemi” sayfa 1013.
- “Bağlantı kesme yöntemi” sayfa 1013.

Özellik Erişimi

Aşağıdaki özelliklere herhangi bir zamanda erişilebilir.

- “AlternateUsertanıtıcı özelliği” sayfa 1004.
- “CompletionCode özelliği” sayfa 1006.
- “ConnectionStatus özelliği” sayfa 1007.
- “ReasonCode özelliği” sayfa 1010.

Kalan özelliklere yalnızca, nesne bir kuyruk yöneticisine bağlıysa ve kullanıcı kimliği o kuyruk yöneticisine ilişkin sorgulama için yetkilendirildiyse erişilebilir. Diğer bir kullanıcı kimliği ayarlandıysa ve yürürlükteki kullanıcı kimliği bu kimliği kullanmaya yetkiliyse, diğer kullanıcı kimliği sorgulamak için yetki olarak denetlenir.

Bu koşullar geçerli değilse, WebSphere MQ Automation Classes for ActiveX , kuyruk yöneticisine bağlanmayı dener ve otomatik olarak sorgulanmak üzere açar. Bu işlem başarısız olursa, çağrı bir CompletionCode olarak MQCC_FAILED ve aşağıdaki ReasonCodes' lerden birini ayarlar:

- MQRC_CONNECTION_BROKEN
- MQRC_NOT_YETKILI
- MQRC_Q_MGR_NAME_ERROR
- MQRC_Q_MGR_NOT_VAR

AlternateUsertanıtıcı özelliği

Okuma-yazma. Kuyruk yöneticisi özniteliklerine erişimi doğrulamak için kullanılacak diğer kullanıcı kimliği.

IsConnected değeri TRUE ise bu özellik ayarlanmamalıdır.

Nesne açık durumdayken bu özellik ayarlanamaz.

Defined in: MQQueueManager sınıfı

Data Type: 12 karakterlik bir dizgi

Syntax: almak için: *altuser $= MQQueueManager*.**AlternateUserId** ayarlamak için: *MQQueueManager*.**AlternateUserId** = *altuser $*

AuthorityEvent özelliği

Salt okunur. MQI AuthorityEvent özniteliği.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *authevent = MQQueueManager*.**AuthorityEvent**

BeginOptions özelliği

Okuma-yazma. Bunlar, Başlat (Begin) yöntemi için geçerli olan seçeneklerdir. Başlangıçta MQBO_NONE.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQBO_NONE

Sözdizimi: almak için: *beginoptions & =MQQueueManager*.**BeginOptions**

Ayarlamak için: *MQQueueManager*.**BeginOptions**=*beginoptions &*

ChannelAutoTanımlaması özelliği

Salt okunur. Bu, otomatik kanal tanımlamasına izin verilip verilmediğini denetler.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCHAD_ENGELLI
- MQCHAD_ENABLED

Sözdizimi: almak için: *channelautodef & = MQQueueManager*.**ChannelAutoTanımlaması**

ChannelAutoDefinitionEvent özelliği

Salt okunur. Bu, otomatik kanal tanımlama olaylarının oluşturulup oluşturulmayacağını denetler.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *channelautodefarvent & =MQQueueManager*.**ChannelAutoDefinitionEvent**

ChannelAutoDefinitionExit özelliği

Salt okunur. Otomatik kanal tanımlaması için kullanılan kullanıcı çıkışısının adı.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Dizgi

Sözdizimi: almak için: *channelautodefexit$= MQQueueManager*.**ChannelAutoDefinitionExit**

CharacterSet özelliği

Salt okunur. MQI CodedCharSetId özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *characterset & = MQQueueManager*.**CharacterSet**

CloseOptions özelliği

Okuma-yazma. Kuyruk yöneticisi kapatıldığında ne olacağını denetlemek için kullanılan seçenekler. İlk değer MQCO_NONE olur.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCO_NONE

Sözdizimi: almak için: *closeopt & = MQQueueManager*.**CloseOptions**

Ayarlamak için: *MQQueueManager*.**CloseOptions** =*closeopt &*

CommandInputQueueName özelliği

Salt okunur. MQI CommandInputQName özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *commandinputqname $= MQQueueManager*.**CommandInputQueueName**

CommandLevel özelliği

Salt okunur. WebSphere MQ kuyruk yöneticisi somutlamasının (MQI CommandLevel özniteliğinin) sürümünü ve düzeyini döndürür.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *level & = MQQueueManager*.**CommandLevel**

CompletionCode özelliği

Salt okunur. Nesneye yönelik olarak verilen son yöntem ya da özellik erişimi tarafından ayarlanan tamamlanma kodunu döndürür.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *completioncode & = MQQueueManager*.**CompletionCode**

ConnectionHandle özelliği

Salt okunur. WebSphere MQ kuyruk yöneticisi nesnesine ilişkin bağlantı tanıtıcısı.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Sözdizimi: almak için: *hconn & = MQQueueManager.***ConnectionHandle**

ConnectionStatus özelliği

Salt okunur. Nesnenin, kuyruk yöneticisine bağlı olup olmadığını gösterir.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *status = MQQueueManager*.**ConnectionStatus**

ConnectOptions özelliği

Okuma yazma. Bu seçenekler Connect yöntemi için geçerlidir. Başlangıçta MQCNO_NONE.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCNO_STANDARD_BINDING
- MQCNO_FASTPATH_BINDING
- MQCNO_NONE

Sözdizimi: almak için: *connectoptions & =MQQueueManager*.**ConnectOptions**

Ayarlamak için: *MQQueueManager*.**ConnectOptions**=*connectoptions &*

DeadLetterQueueName özelliği

Salt okunur. MQI DeadLetterQName özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *dlqname $= MQQueueManager*.**DeadLetterQueueName**

DefaultTransmissionQueueName özelliği

Salt okunur. MQI DefXmitQName özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *defxmiqname $= MQQueueManager*.**DefaultTransmissionQueueName**

Açıklama özelliği

Salt okunur. MQI QMgrDesc özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: 64 karakter dizgisi

Sözdizimi: almak için: *description $= MQQueueManager*.**Açıklama**

DistributionLists özelliği

Salt okunur. Bu, dağıtım listelerini desteklemek için kuyruk yöneticisinin yeteneğidir.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *distributionlist= MQQueueManager*.**DistributionLists**

InhibitEvent özelliği

Salt okunur. MQI InhibitEvent özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *inhibevent & = MQQueueManager*.**InhibitEvent**

IsConnected özelliği

Kuyruk yöneticisinin şu anda bağlı olup olmadığını gösteren bir değer.

Salt okunur.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *isconnected = MQQueueManager*.**IsConnected**

IsOpen özelliği

Kuyruk yöneticisinin sorgu için açık olup olmadığını gösteren bir değer.

Salt okunur.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri Tipi:
Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *IsOpen = MQQueueManager*.**IsOpen**

LocalEvent özelliği

Salt okunur. MQI LocalEvent özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *localevent & = MQQueueManager*.**LocalEvent**

MaximumHandles özelliği

Salt okunur. MQI MaxHandles özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxhandle & = MQQueueManager*.**MaximumHandles**

MaximumMessageUzunluk özelliği

Salt okunur. MQI MaxMsgLength Kuyruk Yöneticisi özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxmessageength & = MQQueueManager*.**MaximumMessageLength**

MaximumPriority özelliği

Salt okunur. MQI MaxPriority özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxpriority & = MQQueueManager*.**MaximumPriority**

MaximumUncommittedİletileri özelliği

Salt okunur. MQI MaxUncommittedMsgs özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxuncommittion & = MQQueueManager*.**MaximumUncommittedİletileri**

Ad özelliği

Okuma-yazma. MQI QMgrName özniteliği. MQQueueManager bağlandıktan sonra bu özellik yazılamaz.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *name $= MQQueueManager*.**name**

Ayarlamak için: *MQQueueManager*.**name** = *name $*

Not: Visual Basic, "Name" özelliğini görsel arabirimde kullanılmak üzere ayırır. Bu nedenle, Visual Basic içinde kullanılırken "name" ("ad") adlı küçük harf kullanımı da kullanılır.

ObjectHandle özelliği

Salt okunur. WebSphere MQ kuyruk yöneticisi nesnesine ilişkin nesne tanıtıcısı.

Tanımlandığı yer:
MQQueueManager sınıfı

Veri türü
Uzun

Sözdizimi: almak için: *hobj & = MQQueueManager.*.**ObjectHandle**

PerformanceEvent özelliği

Salt okunur. MQI PerformanceEvent özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *perfevent & = MQQueueManager*.PerformanceEvent

Platform özelliği

Salt okunur. MQI Platform özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQPL_WINDOWS_NT
- MQPL_WINDOWS

Sözdizimi: almak için: *platform & = MQQueueManager*.**Altyapı**

ReasonCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimiyle belirlenen neden kodunu döndürür.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasoncode & = MQQueueManager*.**ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQQueueManager*.**ReasonName**

RemoteEvent özelliği

Salt okunur. MQI RemoteEvent özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *remoteevent & = MQQueueManager*.**RemoteEvent**

StartStopOlay özelliği

Salt okunur. MQI StartStopolayı özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *strstpevent & = MQQueueManager*.**StartStopOlayı**

SyncPointKullanılabilirlik özelliği

Salt okunur. MQI SyncPoint özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Değerler:

- MQSP_AVALABILIR
- MQSP_NOT_VAR

Sözdizimi: almak için: *syncpointavailability & = MQQueueManager*.**SyncPointUygunluğu**

TriggerInterval özelliği

Salt okunur. MQI TriggerInterval özniteliği.

Tanımlı: MQQueueManager sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *trigint & = MQQueueManager*.**TriggerInterval**

AccessQueue yöntemi

Bir MQQueue nesnesi yaratır ve kuyruğun bağlantı başvurusu özelliğini ayarlayarak bu nesneyi bu MQQueueManager nesnesiyle ilişkilendirir. MQQueue nesnesinin Name (Ad), OpenOptions, DynamicQueueName ve AlternateUsertanıtıcısı özelliklerini sağlanan değerlere ayarlar ve onu açma girişiminde bulunur.

Açma işlemi başarısız olursa, arama başarısız olur. Nesneye karşı bir hata olayı oluşturulur. Nesnenin ReasonCode ve CompletionCodeve MQSession ReasonCode ve CompletionCode değeri belirlenir.

DynamicQueueName, QueueManagerName ve AlternateUsertanıtıcı değiştirgeleri isteğe bağlıdır ve varsayılan değer "" ' dir.

Kuyruk özellikleri okunacaksa, diğer seçeneklere ek olarak OpenOption MQOO_SORGULAMA/ BELIRTILMELI.

Açılacak kuyruk yerelse, QueueManagerAdını ayarlamaz ya da "" değerine ayarlamayın. Tersi durumda, kuyruğun sahibi olan uzak kuyruk yöneticisinin adını ve uzak kuyruğun yerel tanımlamasını açma girişiminde bulunmaya çalışılır. Uzak kuyruk adı çözümlemesi ve kuyruk yöneticisi yöneltmesi hakkında ek bilgi için bkz. Diğer adlar nedir? .

Ad özelliği bir model kuyruğu adı olarak ayarlandıysa, DynamicQueueName$ parametresinde yaratılacak dinamik kuyruk adını belirtin. DynamicQueueName$ parametresinde sağlanan değer "" ise, kuyruk nesnesine ayarlanan ve açık aramada kullanılan değer "AMQ. *" olur. Dinamik kuyrukların adlandırılmasıyla ilgili ek bilgi için “Dinamik kuyruklar oluşturma” sayfa 214 ' e bakın.

Tanımlama

Tanımlanan: MQQueueManager sınıfı.

Sözdizimi

Sözdizimi: set queue = MQQueueManager.**AccessQueue**(Name$, OpenOptions&, QueueManagerName$, DynamicQueueName$, AlternateUserId$)

Parametreler

Name$ Dize. WebSphere MQ kuyruğunun adı.

OpenOptions: Uzun. Kuyruk açıldığında kullanılacak seçenekler. Bkz. OpenOptions (MQUZE).

QueueManagerName$ Dizesi. Açılacak kuyruğun sahibi olan kuyruk yöneticisinin adı. "" değeri, kuyruk yöneticisinin yerel olduğunu belirtir.

*DynamicQueueName$* Dizesi. Name$ parametresi bir model kuyruğunu belirtiyorsa, kuyruğun açıldığı sırada dinamik kuyruğa atanan ad.

AlternateUserId$ Dize. Kuyruğu açarken erişimi doğrulamak için kullanılan diğer kullanıcı kimliği.

AddDistributionListe yöntemi

Yeni bir MQDistributionList nesnesi yaratır ve onun bağlantı başvurusunu sahip kuyruk yöneticisine ayarlar.

Tanımlandığı yer:
MQQueueManager sınıfı

Sözdizimi: *set distributionlist* = **MQQueueManager.***AddDistributionListesi*

Geriletme yöntemi

Kesinleştirilmemiş herhangi bir iletiyi geri alır ve son eşitleme noktasından bu yana bir iş biriminin bir parçası olarak bu işlemi alır.

Tanımlı: MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager.***Backout ()**

Başlangıç yöntemi

Kuyruk yöneticisi tarafından eşgüdümlü bir çalışma birimi başlatır. Başlangıç seçenekleri, bu yöntemin davranışını etkiler.

Tanımlandığı yer:
MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager*.**Begin ()**

ClearErrorCodes yöntemi

CompletionCode öğesini, hem MQQueueManager sınıfı, hem de MQSession sınıfı için MQCC_NONE ve MQRC_NONE MQRC_NONE olarak sıfırlar.

Tanımlı: MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager***.ClearErrorCodes ()**

Kesinleştirme yöntemi

Son eşitleme noktasından bu yana bir iş biriminin bir parçası olarak, herhangi bir ileti koyar ve bunu alır.

Tanımlı: MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager***.Commit ()**

Bağlanma yöntemi

Connects the MQQueueManager object to a real queue manager via the WebSphere MQ MQI client or server. Bu yöntem, bağlantının yanı sıra, kuyruk yöneticisi nesnesini de açar; böylece, bu nesne sorgulanabilir.

IsConnected değerini TRUE olarak ayarlar.

Bir kuyruk yöneticisine bağlanmasına izin verilen ActiveX başına en çok bir MQQueueManager nesnesi kullanılmasına izin verilir.

Tanımlı: MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager***.Connect ()**

Bağlantı kesme yöntemi

Kuyruk yöneticisinden MQQueueManager nesnesinin bağlantısını keser.

IsConnected öğesini FALSE olarak ayarlar.

MQQueueManager nesnesiyle ilişkili tüm kuyruk nesneleri kullanılamaz duruma gelir ve yeniden açılamaz.

Kesinleştirilmemiş değişiklikler (ileti koyar ve alır) kesinleştirilir.

Tanımlı: MQQueueManager sınıfı

Sözdizimi: Call *MQQueueManager***.Bağlantıyı Kes ()**

MQQueue sınıfı

Bu sınıf, bir WebSphere MQ kuyruğuna erişimi temsil eder. Bu bağlantı, ilişkili bir MQQueueManager nesnesi tarafından sağlanıyor. Bu sınıfların bir nesnesi yok edildiğinde otomatik olarak kapatılır.

Bulundurma

MQQueue sınıfı MQQueueManager sınıfı tarafından içerilir.

Yaratma

New , yeni bir MQQueue nesnesi yaratır ve tüm özellikleri ilk değerlere ayarlar. Diğer bir seçenek olarak, MQQueueManager sınıfının AccessQueue yöntemini kullanın.

Sözdizimi

```
Dim que As New MQQueue Set que  = New MQQueue
```

Özellikler

- “OpenOptions özelliği” sayfa 1022.
- “OpenOutputCount özelliği” sayfa 1022.
- “OpenStatus özelliği” sayfa 1022.
- “ProcessName özelliği” sayfa 1023.
- “QueueManagerAd özelliği” sayfa 1023.
- “QueueType Özelliği” sayfa 1023.
- “ReasonCode özelliği” sayfa 1023.
- “ReasonName özelliği” sayfa 1023.
- “RemoteQueueManagerName özelliği” sayfa 1023.
- “RemoteQueueAd özelliği” sayfa 1024.
- “ResolvedQueueManagerName özelliği” sayfa 1024.
- “ResolvedQueueAd özelliği” sayfa 1024.
- “RetentionInterval özelliği” sayfa 1024.
- “Kapsam özelliği” sayfa 1024.
- “ServiceInterval özelliği” sayfa 1024.
- “ServiceIntervalOlay özelliği” sayfa 1025.
- “Paylaşılabilirlik özelliği” sayfa 1025.
- “TransmissionQueueAd özelliği” sayfa 1025.
- “TriggerControl özelliği” sayfa 1025.
- “TriggerData özelliği” sayfa 1025.
- “TriggerDepth özelliği” sayfa 1026.
- “TriggerMessageÖnceliği özelliği” sayfa 1026.
- “TriggerType özelliği” sayfa 1026.
- “Kullanım özelliği” sayfa 1026.

Yöntemler

- “ClearErrorCodes yöntemi” sayfa 1026
- “Yöntemi kapat” sayfa 1027
- “Get yöntemi” sayfa 1027
- “Open yöntemi” sayfa 1027
- “Put yöntemi” sayfa 1028

Özellik Erişimi

Kuyruk nesnesi bir kuyruk yöneticisine bağlanmadıysa, aşağıdaki özellikleri okuyabilirsiniz:

- “CompletionCode özelliği” sayfa 1017
- “OpenStatus özelliği” sayfa 1022
- “ReasonCode özelliği” sayfa 1023

ve okuma yazma işlemi yapabilmeniz için aşağıdakileri yapabilirsiniz:

- “AlternateUsertanıtıcı özelliği” sayfa 1016
- “CloseOptions özelliği” sayfa 1017
- “ConnectionReference özelliği” sayfa 1017
- “Ad özelliği” sayfa 1021
- “OpenOptions özelliği” sayfa 1022

Kuyruk nesnesi kuyruk yöneticisine bağlıysa, tüm özellikleri okuyabilirsiniz.

Kuyruk Özniteliği özellikleri

Önceki bölümde listelenmeyen özellikler, temeldeki WebSphere MQ kuyruğunda yer alan tüm özniteliklerdir. Bunlar yalnızca, nesne bir kuyruk yöneticisine bağlı olduğunda ve kullanıcının kullanıcı kimliği sorgulamak ya da bu kuyruğa karşı ayarlama için yetkilendirilmiş ise erişilebilir. Başka bir kullanıcı kimliği belirlendiyse ve yürürlükteki kullanıcı kimliği bu kimliği kullanmaya yetkiliyse, bunun yerine diğer kullanıcı kimliği yetki denetlemesi için işaretlendi.

Özellik, verili QueueTypeiçin uygun bir özellik olmalıdır. Ek bilgi için Kuyruklar için öznitelikler başlıklı konuya bakın.

Bu koşullar geçerli değilse, özellik erişimi bir CompletionCode olarak MQCC_FAILED ve aşağıdaki ReasonCodes' lardan biri olarak ayarlanacaktır:

- MQRC_CONNECTION_BROKEN
- MQRC_NOT_YETKILI
- MQRC_Q_MGR_NAME_ERROR
- MQRC_Q_MGR_NOT_CONNECTED
- MQRC_SELECTOR_NOT_FOR_TYPE (CompletionCode , MQCC_UYARI olur)

Kuyruğun Açılması

Bir MQQueue nesnesini yaratmanın tek yolu, MQQueueManager AccessQueue yöntemini ya da Yeni öğesini kullanarak olur. Açık bir MQQueue nesnesi, kapatılıncaya ya da silininceye ya da kuyruk yöneticisi nesnesi silininceye ya da kuyruk yöneticisine bağlantı kayboluncaya kadar açık kalır (OpenStatus= TRUE). MQQueue CloseOptions özelliğinin değeri, MQQueue nesnesi silindiğinde gerçekleşen kapatma işleminin işleyişini denetler.

The MQQueueManager AccessQueue method opens the queue using the OpenOptions parameter. The MQQueue.Open method opens the queue using the OpenOptions property. WebSphere MQ , açık kuyruk sürecinin bir parçası olarak kullanıcı yetkilendirmesine karşı OpenOptions ' ı doğrular.

AlternateUsertanıtıcı özelliği

Okuma-yazma. Açıldığı sırada kuyruğa erişimi doğrulamak için kullanılan diğer kullanıcı kimliği.

Bu özellik, nesne açıkken ayarlanamaz (bu durumda, IsOpen TRUE olduğunda).

Tanımlı: MQQueue sınıfı

Veri Tipi: 12 karakter dizgisi

Sözdizimi: almak için: *altuser $= MQQueue***.AlternateUserTnt**

Ayarlamak için: *MQQueue.***AlternateUserId** = *altuser $*

BackoutRequeueAd özelliği

Salt okunur. MQI BackOutRequeueQName özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *backoutrequiuename $= MQQueue***.BackoutRequeueAd**

BackoutThreshold

Salt okunur. MQI BackoutThreshold özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. BackoutThreshold (MQUZE)

Sözdizimi: almak için: *backoutthreshold & = MQQueue.***BackoutThreshold**

BaseQueueAd özelliği

Salt okunur. Diğer adın çözümleyicilerin bulunduğu kuyruk adı.

Yalnızca diğer ad kuyrukları için geçerlidir.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *baseqname $= MQQueue***.BaseQueueName**

CloseOptions özelliği

Okuma yazma. Kuyruk kapatıldığında ne olacağını denetlemek için kullanılan seçenekler.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQCO_NONE
- MQCO_DELETE
- MQCO_DELETE_PURGE

MQCO_DELETE ve MQCO_DELETE_PURGE yalnızca dinamik kuyruklar için geçerlidir.

Sözdizimi: almak için: *closeopt & = MQQueue***.CloseOptions**

Ayarlamak için: *MQQueue***.CloseOptions** = *closeopt &*

CompletionCode özelliği

Salt okunur. Nesneye yönelik olarak verilen son yöntem ya da özellik erişimi tarafından ayarlanan tamamlanma kodunu döndürür.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *completioncode & = MQQueue***.CompletionCode**

ConnectionReference özelliği

Okuma-yazma. Bir kuyruk nesnesinin ait olduğu kuyruk yöneticisi nesnesini tanımlar. Bir kuyruk açıkken bağlantı başvurusu yazılamaz.

Tanımlı: MQQueue sınıfı

Veri Tipi: MQQueueManager

Değerler:

- Etkin bir WebSphere MQ Queue Manager nesnesine başvuru

Sözdizimi: Ayarlamak Için: *set MQQueue***.ConnectionReference** = *ConnectionReference*

Almak için: *set ConnectionReference = MQQueue*.**ConnectionReference**

CreationDateSaat özelliği

Salt okunur. Bu kuyruğun yaratıldığı tarih ve saat.

Tanımlı: MQQueue sınıfı

Veri Tipi: Tip 7 (tarih/saat) ya da EMPTY tipinde olan değişken

Sözdizimi: almak için: *datetime = MQQueue*.**CreationDateTime**

CurrentDepth özelliği

Salt okunur. Şu anda kuyruklardaki ileti sayısı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *currentdepth & = MQQueue*.**CurrentDepth**

DefaultInputOpenOption özelliği

Salt okunur. OpenOptions MQOO_INPUT_AS_Q_DEF değerini belirtiyorsa, kuyruğun açılacağı yolu denetler.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_SHARED

Sözdizimi: almak için: *defaultinop & = MQQueue*.**DefaultInputOpenOption**

DefaultPersistence özelliği

Salt okunur. Kuyruklardaki iletiler için varsayılan kalıcılık.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *defpersistence & = MQQueue*.**DefaultPersistence**

DefaultPriority özelliği

Salt okunur. Kuyruklardaki iletiler için varsayılan öncelik değeri.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *defpriority & = MQQueue*.**DefaultPriority**

DefinitionType özelliği

Salt okunur. Kuyruk tanımlaması tipi.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQDT_ÖNCEDEN tanımlı
- MQQDT_PERMANENT_DYNAMIC

- MQQDT_TEMPORARY_DYNAMIC

Sözdizimi: almak için: *deftype & = MQQueue*.**DefinitionType**

DepthHighOlay özelliği

Salt okunur. MQI QDepthHighOlay özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *depthhighevent & = MQQueue.*.**DepthHighOlayı**

DepthHighSınır özelliği

Salt okunur. MQI QDepthHighSınır özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *depthhighlimit & = MQQueue.*.**DepthHighLimit**

DepthLowOlay özelliği

Salt okunur. MQI QDepthLowOlay özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *depthlowevent & = MQQueue.*.**DepthLowOlayı**

DepthLowÖzelliği sınırla

Salt okunur. MQI QDepthLowSınırı özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *depthlowlimit & = MQQueue.*.**DepthLowSınır**

DepthMaximumOlay özelliği

Salt okunur. MQI QDepthMaxOlay özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQEVR_DISABLE
- MQEVRENABLED

Sözdizimi: almak için: *depthmaximument & = MQQueue.*.**DepthMaximumOlayı**

Açıklama özelliği

Salt okunur. Kuyruğun açıklaması.

Tanımlı: MQQueue sınıfı

Veri Tipi: 64 karakter dizgisi

Sözdizimi: almak için: *description $= MQQueue*.**Açıklama**

DynamicQueueAd özelliği

Okuma yazma, kuyruk açık olduğunda salt okunur olur.

Bu, bir model kuyruğu açıldığında kullanılan dinamik kuyruk adını denetler. Kullanıcı bir genel arama karakteri ile bir özellik kümesi (yalnızca kuyruk kapatıldığında) ya da MQQueueManager.AccessQueue() için bir parametre olarak ayarlanabilir.

Dinamik kuyruğun gerçek adı, QueueNamesorgulanarak bulunur.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Değerler:

- Geçerli bir WebSphere MQ kuyruk adı.

Sözdizimi: Ayarlamak Için: *MQQueue*.**DynamicQueueName** = *dynamicqueuename $*

Almak için: *dynamicqueuename $ = MQQueue*.**DynamicQueueAd**

HardenGetGeri alma özelliği

Salt okunur. Doğru bir geri sayım sayımının sağlanıp korunmayacağı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQA_BACKUT_HARDENED
- MQQA_BACKUTUP_NOT HARDENED

Sözdizimi: Alınmak için: *hardengetback & = MQQueue*.**HardenGetBackout**

InhibitGet özelliği

Okuma-yazma. MQI InhibitGet özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQA_GET_INHIBITED
- MQQA_GET_ALLOWD

Sözdizimi: Alınmak için: *getstatus & = MQQueue*.**InhibitGet**

Ayarlamak için: *MQQueue*.**InhibitGet** = *getstatus &*

InhibitPut özelliği

Okuma-yazma. MQI InhibitPut özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQA_PUT_INHIBITED
- MQQA_PUT_ALLOWD

Sözdizimi: almak için: *putstatus & = MQQueue*.**InhibitPut**

Ayarlamak için: *MQQueue*.**InhibitPut** = *putstatus &*

InitiationQueueAd özelliği

Salt okunur. Başlatma kuyruğunun adı.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *initqname $= MQQueue*.**InitiationQueueName**

IsOpen özelliği

Kuyruğun açık olup olmadığını döndürür.

Salt okunur.

Tanımlı: MQQueue sınıfı

Veri Tipi: Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *open = MQQueue*.**IsOpen**

MaximumDepth özelliği

Salt okunur. Kuyruk derinliği üst sınırı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxdepth & = MQQueue*.**MaximumDepth**

MaximumMessageUzunluk özelliği

Salt okunur. Bu kuyruk için bayt cinsinden izin verilen ileti uzunluğu üst sınırı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *maxmlength & = MQQueue*.**MaximumMessageLength**

MessageDeliverySıra özelliği

Salt okunur. Mesaj teslim sırası.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQMDS_PRIORITY
- MQMDS_FIFO

Sözdizimi: almak için: *messdelseq & = MQQueue*.**MessageDeliverySequence**

Ad özelliği

Okuma-yazma. MQI Kuyruğu özniteliği. Bu özellik, MQQueue açıldıktan sonra yazılamaz.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *name $= MQQueue***.name**

Ayarlamak için: *MQQueue***.name** = *name $*

Not: Visual Basic, "Name" özelliğini görsel arabirimde kullanılmak üzere ayırır. Bu nedenle, Visual Basic kullanirken "name" (alt harf) kullanan Visual Basic kullanimi kullanlmssa.

ObjectHandle özelliği

Salt okunur. WebSphere MQ kuyruk nesnesine ilişkin nesne tanıtıcısı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *hobj & = MQQueue.***ObjectHandle**

OpenInputSayı özelliği

Salt okunur. Giriş için açılan açma sayısı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *openincount & = MQQueue***.OpenInputCount**

OpenOptions özelliği

Okuma-yazma. Kuyruğu açmak için kullanılacak seçenekler.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. OpenOptions (MQUZE).

Sözdizimi: almak için: *openopt & = MQQueue***.OpenOptions**

Ayarlamak için: *MQQueue.***OpenOptions** = *openopt &*

OpenOutputCount özelliği

Salt okunur. Çıkış için açılan açma sayısı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *openoutcount & = MQQueue***.OpenOutputCount**

OpenStatus özelliği

Salt okunur. Kuyruğun açılıp açılmayaca § edilmediğini belirtir. İlk değer, AccessQueue yönteminden sonra TRUE ya da New 'den sonra FALSE değerini verir.

Tanımlı: MQQueue sınıfı

Veri Tipi: Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *status & = MQQueue***.OpenStatus**

ProcessName özelliği

Salt okunur. MQI ProcessName özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *procname $ =MQQueue***.ProcessName**

QueueManagerAd özelliği

Okuma-yazma. WebSphere MQ kuyruk yöneticisi adı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Dize

Sözdizimi: almak için: *QueueManagerName$ = MQQueue***.QueueManagerName**

Ayarlamak için: *MQQueue***.QueueManagerName** = *QueueManagerName$*

QueueType Özelliği

Salt okunur. MQI QType özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQT_ALIAS
- MQQT_LOCAL
- MQQT_MODEL
- MQQT_REMOTE

Sözdizimi: almak için: *queuetype & = MQQueue***.QueueType**

ReasonCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimiyle belirlenen neden kodunu döndürür.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasoncode & = MQQueue***.ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Tanımlı: MQQueue sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQQueue***.ReasonName**

RemoteQueueManagerName özelliği

Salt okunur. Uzak kuyruk yöneticisinin adı. Yalnızca uzak kuyruklar için geçerlidir.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *remqmanname $= MQQueue*.**RemoteQueueManagerName**

RemoteQueueAd özelliği

Salt okunur. Uzak kuyruk yöneticisininde bilindiği gibi, kuyruğun adı. Yalnızca uzak kuyruklar için geçerlidir.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *remqname $= MQQueue*.**RemoteQueueName**

ResolvedQueueManagerName özelliği

Salt okunur. Yerel kuyruk yöneticisiniyle bilinen son hedef kuyruk yöneticisinin adı.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *resqmanname $= MQQueue*.**ResolvedQueueManagerName**

ResolvedQueueAd özelliği

Salt okunur. Yerel kuyruk yöneticisiniyle bilinen son hedef kuyruğunun adı.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *resqname $= MQQueue*.**ResolvedQueueAdı**

RetentionInterval özelliği

Salt okunur. Kuyruğun alıkonması gereken süre.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *retinterval & = MQQueue*.**RetentionInterval**

Kapsam özelliği

Salt okunur. Bu kuyruğa ilişkin bir girişin bir hücre dizininde de bulunup bulunmadığını denetler.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQSCO_Q_MGR
- MQSCO_CEL

Sözdizimi: almak için: *scope & = MQQueue*.**Scope**

ServiceInterval özelliği

Salt okunur. MQI QServiceInterval özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *serviceinterval & = MQQueue.***ServiceInterval**

ServiceIntervalOlay özelliği

Salt okunur. MQI QServiceIntervalOlay özniteliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQSI_YüKSEK
- MQQSI_OK
- MQQSI_NONE

Sözdizimi: almak için: *serviceintervalevent & = MQQueue.***ServiceIntervalOlayı**

Paylaşılabilirlik özelliği

Salt okunur. Kuyruk paylaşılabilirliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQQA_SHAREABLE
- MQQA_NOT_SHAREABLE

Sözdizimi: almak için: *shareubility & = MQQueue***.Sharelability**

TransmissionQueueAd özelliği

Salt okunur. İletim kuyruğu adı. Yalnızca uzak kuyruklar için geçerlidir.

Tanımlı: MQQueue sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Almak Için: *transqname $= MQQueue***.TransmissionQueueAdı**

TriggerControl özelliği

Okuma-yazma. Tetik kontrolü.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQTC_OFF
- MQTC_ON

Sözdizimi: Almak Için: *trigcontrol & = MQQueue***.TriggerControl**

Ayarlamak için: *MQQueue***.TriggerControl** = *trigcontrol &*

TriggerData özelliği

Okuma-yazma. Verileri tetikler.

Tanımlı: MQQueue sınıfı

Veri Tipi: 64 karakter dizgisi

Sözdizimi: Almak Için: *trigdata $= MQQueue***.TriggerData**

Ayarlamak için: *MQQueue*.**TriggerData** = *trigdata $*

TriggerDepth özelliği

Okuma-yazma. Bir tetikleme iletisi yazılmadan önce kuyruğun üzerinde olması gereken ileti sayısı.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: Almak Için: *trigdepth & = MQQueue*.**TriggerDepth**

Ayarlamak için: *MQQueue*.**TriggerDepth** = *trigdepth &*

TriggerMessageÖnceliği özelliği

Okuma-yazma. Tetikleyiciler için eşik iletisi önceliği.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Sözdizimi: Almak Için: *trigmesspriority & = MQQueue*.**TriggerMessageÖnceliği**

Ayarlamak için: *MQQueue*.**TriggerMessagePriority** = *trigmesspriority &*

TriggerType özelliği

Okuma-yazma. Tetikleyici tipi.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQTT_NONE
- MQTT_BIRINCI
- MQTT_EVERY
- MQTT_DERINLIK

Sözdizimi: Almak Için: *trigtype & = MQQueue*.**TriggerType**

Ayarlamak için: *MQQueue*.**TriggerType** = *Trigtype &*

Kullanım özelliği

Salt okunur. Kuyruğun ne için kullanıldığını gösterir.

Tanımlı: MQQueue sınıfı

Veri Tipi: Uzun

Değerler:

- MQUS_NORMAL
- MQUS_ILETIMI

Sözdizimi: almak için: *usage & = MQQueue*.**Kullanım**

ClearErrorCodes yöntemi

CompletionCode 'u MQCC_OK ve ReasonCode ' yi hem MQQueue sınıfı, hem de MQSession sınıfı için MQRC_NONE olarak sıfırlar.

Tanımlı: MQQueue sınıfı

Sözdizimi: Call *MQQueue*.**ClearErrorCodes ()**

Yöntemi kapat

CloseOptions' un geçerli değerlerini kullanarak bir kuyruğu kapatır.

Tanımlı: MQQueue sınıfı

Sözdizimi: Call *MQQueue*.**Kapat ()**

Get yöntemi

Kuyruktan bir ileti alır.

Bu yöntem, nesne olarak bir MQMessage nesnesini, nesnenin MQMD ' deki bazı alanları giriş değiştirgeleri olarak kullanarak alır. Özellikle, MessageId ve CorrelId alanları kullanılır; bu nedenle, bu alanların gerektiği şekilde ayarlandığından emin olun. Bu alanlarla ilgili daha fazla bilgi için bkz. MsgId (MQBYTE24) ve CorrelId (MQBYTE24) .

Yöntem başarısız olursa, MQMessage nesnesi değişmeden kalır. Başarılı olursa, MQMessage nesnesinin MQMD ve Message Data kısımlarının yerini, gelen iletiden MQMD ve İleti Verileri alır. MQMessage denetim özellikleri aşağıdaki gibi ayarlandı:

- **MessageLength** , WebSphere MQ iletisinin uzunluğuna ayarlanır
- **DataLength** , WebSphere MQ iletisinin uzunluğuna ayarlanır.
- **DataOffset** sıfır olarak ayarlandı

Tanımlandığı yer:
MQQueue sınıfı

Sözdizimi: Call *MQQueue*.**Get**(*İleti, GetMsgSeçenekleri, GetMsgUzunluk*)

Parametreler
İleti:

Alınacak iletiyi gösteren MQMessage nesnesi.

GetMsgSeçenekleri:

Alma işlemini denetlemek için isteğe bağlı MQGetMessageSeçenekleri nesnesi. Bu parametre belirlenmezse, varsayılan MQGetMessageSeçenekleri kullanılır.

GetMsgUzunluğu:

Kuyruktan alınan WebSphere MQ iletisinin uzunluk üst sınırını denetlemek için isteğe bağlı 2 ya da 4 baytlık uzunluk değeri.

MQGMO_ACCEPT_TRUNCATED_MSG seçeneği belirtilirse, GET işlemi MQCC_UYARI tamamlanma koduyla başarılı olur ve ileti büyüklüğü belirtilen uzunluğu aşarsa MQRC_TRUNCATED_MSG_ACCEPTED neden kodu kabul edilir.

MessageData , ilk GetMsgbayt veri byte 'ı içerir.

MQGMO_ACCEPT_TRUNCATED_MSG ***is not*** belirtilirse ve ileti büyüklüğü belirtilen uzunluğu aşarsa, MQCC_FAILED işleminin tamamlanma kodu bir arada MQRC_TRUNCATED_MESSAGE_FAILED dönüş koduyla birlikte başarısız oldu.

İleti arabelleğindeki içerik tanımsız olursa, toplam ileti uzunluğu, alınan iletinin tam uzunluğuna ayarlanır.

İleti uzunluğu parametresi belirlenmezse, ileti arabelleğindeki uzunluk, gelen iletinin en az büyüklüğüne göre otomatik olarak ayarlanır.

Open yöntemi

Şu anki değerleri kullanarak bir kuyruk açar:

1. QueueName
2. QueueManagerAdı

3. AlternateUserTanıtıcısı
4. DynamicQueueAdı

Tanımlandığı yer:
MQQueue sınıfı

Sözdizimi: Call *MQQueue***.Open ()**

Put yöntemi

Kuyruğa ileti yerleştirir.

Bu yöntem, bir MQMessage nesnesini değiştirge olarak alır. Bu nesnenin İleti Tanımlayıcısı (MQMD) özellikleri bu yöntemin bir sonucu olarak değiştirilebilir. Bu yöntem çalıştırıldıktan hemen sonra sahip oldukları değerler, WebSphere MQ' ya konulan değerlerdir.

Koyma işlemi tamamlandıktan sonra MQMessage nesnesinde yapılan değişiklikler, WebSphere MQ kuyruğunda gerçek iletiyi etkilemez.

Tanımlandığı yer:
MQQueue sınıfı

Sözdizimi: Call *MQQueue***.put**(*İleti, PutMsgSeçenekleri)*

Parametreler
İleti

Konmak için iletiyi gösteren MQMessage nesnesi.

PutMsgSeçenekleri

Koyma işlemini denetleyen seçenekleri içerenMQPutMessageSeçenekleri nesnesi. Bu seçenek belirlenmezse, varsayılan PutMessage(PutMessage) seçenekleri kullanılır.

MQMessage sınıfı

Bu sınıf bir WebSphere MQ iletisini gösterir. WebSphere MQ ileti tanımlayıcısını (MQMD) kapsüllemek için özellikleri içerir ve uygulama tarafından tanımlanan ileti verilerini tutmak için bir arabellek sağlar.

Sınıf, bir ActiveX uygulamasından bir MQMessage nesnesine veri kopyalamak için Yazma yöntemlerini içerir. Benzer şekilde, sınıf bir MQMessage nesnesindeki verileri bir ActiveX uygulamasına kopyalamak için Okuma yöntemlerini içerir. Sınıf, arabelleğe otomatik olarak ayrılan bellek ayırma ve serbest bırakma işlemlerini yönetir. Bir MQMessage nesnesi yaratıldığında arabellek büyüklüğünü bildirmek zorunda değildir; arabellek bu nesneye yazılan verileri barındıracak şekilde büyür.

Arabellek büyüklüğü o kuyruğun MaximumMessageLength özelliğini aşarsa, bir iletiyi WebSphere MQ kuyruğuna yerleştiremezsiniz.

Oluşturulduktan sonra, bir MQMessage nesnesi MQQueue.Put yöntemini kullanarak bir WebSphere MQ kuyruğuna konabilir. Bu yöntem, nesnenin MQMD ' nin ve ileti veri bölümlerinin bir kopyasını alır ve kuyruğun üzerine kopyalayan yerlerini alır. Bu nedenle, uygulama, WebSphere MQ kuyruğunda iletiyi etkilemeden bir MQMessage nesnesini değiştirebilir ya da silebilir. Kuyruk yöneticisi, iletiyi WebSphere MQ kuyruğuna kopyalarken, MQMD içindeki bazı alanları ayarlayabilirler.

Gelen bir ileti, MQQueue.Get yöntemini kullanarak bir MQMessage nesnesine okunabilir. Bu işlem, gelen iletiden değerlerle MQMessage nesnesinde önceden bulunan MQMD ya da ileti verilerinin yerini alır. Bu, MQMessage nesnesinin veri arabelleğindeki büyüklüğünü, gelen ileti verilerinin büyüklüğünün eşleştirmek üzere ayarlar.

Bulundurma

İletiler MQSession sınıfı tarafından içerilir.

Yaratma

Yeni , bir MQMessage nesnesi yaratır. İleti tanımlayıcı özellikleri başlangıçta varsayılan değerlere ayarlanır ve İleti Verileri arabelleği boş olur.

Sözdizimi

```
Dim msg As New MQMessage or Set msg = New MQMessage
```

Özellikler

Denetim özellikleri şunlardır:

- “CompletionCode özelliği” sayfa 1031
- “DataLength özelliği” sayfa 1031
- “DataOffset özelliği” sayfa 1031
- “MessageLength özelliği” sayfa 1032
- “ReasonCode özelliği” sayfa 1032
- “ReasonName özelliği” sayfa 1032

İleti tanımlayıcı özellikleri şunlardır:

- “AccountingToken özelliği” sayfa 1032
- “AccountingTokenHex özelliği” sayfa 1033
- “ApplicationIdData özelliği” sayfa 1033
- “ApplicationOriginVeri özelliği” sayfa 1033
- “BackoutCount özelliği” sayfa 1033
- “CharacterSet özelliği” sayfa 1034
- “CorrelationId özelliği” sayfa 1034
- “CorrelationIdHex özelliği” sayfa 1034
- “Kodlama özelliği” sayfa 1035
- “Süre bitimi özelliği” sayfa 1035
- “Feedback özelliği” sayfa 1036
- “Biçim özelliği” sayfa 1036
- “GroupId özelliği” sayfa 1036
- “GroupIdHex özelliği” sayfa 1036
- “MessageData özelliği” sayfa 1037
- “MessageFlags özelliği” sayfa 1037
- “MessageId özelliği” sayfa 1037
- “MessageIdHex özelliği” sayfa 1037
- “MessageSequenceSayı özelliği” sayfa 1038
- “MessageType özelliği” sayfa 1038
- “Göreli konum özelliği” sayfa 1038
- “OriginalLength özelliği” sayfa 1038
- “Kalıcılık özelliği” sayfa 1039
- “Öncelik özelliği” sayfa 1039
- “PutApplicationAd özelliği” sayfa 1039
- “PutApplicationTip özelliği” sayfa 1039

- “PutDateSaat özelliği” sayfa 1039
- “ReplyToQueueManagerAd özelliği” sayfa 1040
- “ReplyToQueueName özelliği” sayfa 1040
- “Rapor özelliği” sayfa 1040
- “TotalMessageUzunluk özelliği” sayfa 1040
- “UserId özelliği” sayfa 1040

Yöntemler

- “ClearErrorCodes yöntemi” sayfa 1041
- “ClearMessage yöntemi” sayfa 1041
- “Okuma yöntemi” sayfa 1041
- “ReadBoolean yöntemi” sayfa 1041
- “ReadByte yöntemi” sayfa 1041
- “ReadDecimal2 yöntemi” sayfa 1042
- “ReadDecimal4 yöntemi” sayfa 1042
- “ReadDouble yöntemi” sayfa 1042
- “ReadDouble4 yöntemi” sayfa 1042
- “ReadFloat yöntemi” sayfa 1042
- “ReadInt2 yöntemi” sayfa 1043
- “ReadInt4 yöntemi” sayfa 1043
- “ReadLong yöntemi” sayfa 1043
- “ReadNullTerminatedString yöntemi” sayfa 1043
- “ReadShort yöntemi” sayfa 1043
- “ReadString yöntemi” sayfa 1044
- “ReadUInt2 yöntemi” sayfa 1044
- “ReadUnsignedByte yöntemi” sayfa 1044
- “ReadUTF yöntemi” sayfa 1044
- “ResizeBuffer yöntemi” sayfa 1045
- “Yazma yöntemi” sayfa 1045
- “WriteBoolean yöntemi” sayfa 1045
- “WriteByte yöntemi” sayfa 1045
- “WriteDecimal2 yöntemi” sayfa 1046
- “WriteDecimal4 yöntemi” sayfa 1046
- “WriteDouble yöntemi” sayfa 1046
- “WriteDouble4 yöntemi” sayfa 1046
- “WriteFloat yöntemi” sayfa 1047
- “WriteInt2 yöntemi” sayfa 1047
- “WriteInt4 yöntemi” sayfa 1047
- “WriteLong yöntemi” sayfa 1047
- “WriteNullTerminatedString yöntemi” sayfa 1047
- “WriteShort yöntemi” sayfa 1048
- “WriteString yöntemi” sayfa 1048
- “WriteUInt2 yöntemi” sayfa 1048

- “WriteUnsignedByte yöntemi” sayfa 1048
- “WriteUTF yöntemi” sayfa 1049

Özellik erişimi

Tüm özellikler her zaman okunabilir.

Denetim özellikleri, okuma-yazma olan DataOffset dışında salt okunurdur. İleti tanımlayıcı özellikleri, yalnızca okunur olan BackoutCount ve TotalMessageUzunluğu dışında, tüm okuma-yazma özellikleridir.

Ancak, ileti bir WebSphere MQ kuyruğuna konduğunda, MQMD özelliklerinin kuyruk yöneticisi tarafından değiştirilebileceğini göz önünde bulundurun. Bunların nasıl değiştirilebileceğiyle ilgili ayrıntılar için MQMD içindeki alanlara bakın.

Veri dönüştürme

İkili verileri bir WebSphere MQ iletisine geçirebilirsiniz; CharacterSet özelliğini, kuyruk yöneticisinin (MQCCSI_Q_MGR) kodlanmış karakter takımı tanıtıcısına (MQCCSI_Q_MGR) ayarlayıp bir dizgi geçirebilirsiniz. chr $işlevini, karakter olmayan verileri dizgiye ayarlamak için kullanabilirsiniz.

Okuma ve Yazma yöntemleri, veri dönüştürmeyi gerçekleştirir. They convert between the ActiveX internal formats, and the WebSphere MQ message formats as defined by the Encoding and CharacterSet properties from the message descriptor. Bir ileti yazarken, bir Yazma yöntemi yayınlamadan önce iletinin alıcısının karakteristiğiyle eşleşen Kodlamaya ve CharacterSet ' e değer ayarlayın. Bir ileti okunurken, bu değerler, gelen MQMD ' de bu değerler belirlendiği için olağan durumda zorunlu değildir.

Bu, MQQueue.Get yöntemi tarafından gerçekleştirilen dönüştürmenin ardından gerçekleşen ek bir veri dönüştürme adımdır.

CompletionCode özelliği

Salt okunur. Bu nesneye yönelik olarak yayınlanan en son yöntem ya da özellik erişimi ile ayarlanan WebSphere MQ tamamlanma kodunu döndürür.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *completioncode & = MQMessage*.**CompletionCode**

DataLength özelliği

Salt okunur. Bu özellik şu değeri döndürür:

```
MQMessage.MessageLength - MQMessage.DataOffset
```

Bir Okuma yönteminden önce, arabellekta beklenen karakter sayısının gerçekte var olup olmadığını denetlemek için kullanılabilir.

Başlangıç değeri sıfır.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *bytessol & = MQMessage*.**DataLength**

DataOffset özelliği

Okuma-yazma. İleti nesnesinin İleti Verileri bölümü içindeki yürürlükteki konum.

Değer, ileti veri arabelleğinin başlangıcındaki byte göreli konumu olarak ifade edilir; arabelleğindeki ilk karakter, sıfır değerinin DataOffset değerine karşılık gelir.

A read or write method commences its operation at the character referenced by DataOffset. Bu yöntemler, arabelleğindeki verileri bu konumdan başlayarak sırayla işler ve son işlenen son işlemden hemen sonra (varsa) byte 'ı (varsa) işaret edecek şekilde DataOffset güncellemesine bakın.

DataOffset , yalnızca sıfır ile MessageLength aralığındaki değerleri kapsayılabilir. DataOffset = MessageLength sona doğru gösterdiğinde, arabelleğin ilk geçersiz karakteridir. Bu durumda yazma yöntemlerine izin verilir; arabelleğdeki verileri genişletir ve MessageLength değerini, eklenen bayt sayısını artırır. Arabelleğin sona ermesinin ötesinde okuma değeri geçerli değil.

Başlangıç değeri sıfır.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *curcpos & = MQMessage*.**DataOffset**

Ayarlamak için: *MQMessage*.**DataOffset** = *curcupos &*

MessageLength özelliği

Salt okunur. DataOffsetdeğerinden bağımsız olarak, karakter cinsinden ileti nesnesinin İleti Verileri bölümünün toplam uzunluğunu döndürür.

Başlangıç değeri sıfır. Bu ileti nesnesine gönderme yapan bir alma yöntemi çağrısından sonra gelen İleti Uzunluğu olarak ayarlanır. Uygulama nesneye veri eklemek için bir Yazma yöntemi kullanıyorsa, bu değer artırılır. Okuma yöntemlerinden etkilenmez.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: Almak Için: *msglength & = MQMessage*.**MessageLength**

ReasonCode özelliği

Salt okunur. Bu nesneye yönelik olarak verilen en son yöntem ya da özellik erişimi tarafından ayarlanan neden kodunu döndürür.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasoncode & = MQMessage*.**ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".
Tanımlı: MQMessage sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQMessage*.**ReasonName**

AccountingToken özelliği

Okuma-yazma. MQMD AccountingToken -İleti Kimliği Bağlamı 'nın bir parçası.

Başlangıç değeri tüm boş değerlerde olur.

Tanımlı: MQMessage sınıfı

Veri Tipi: 32 karakterlik dize

Sözdizimi: almak için: *actoken $= MQMessage*.**AccountingToken**

Ayarlamak için: *MQMessage.***AccountingToken** = *actoken $*

AccountingToken özelliği yerine AccountingTokenHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için “İleti Tanımlayıcısı özellikleri” sayfa 993 konusuna bakın.

AccountingTokenHex özelliği

Okuma-yazma. MQMD AccountingToken -İleti Kimliği Bağlamı 'nın bir parçası.

Her iki karakter, tek bir ASCII karakterinin onaltılık değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çifti tek karakteri "B", vb. temsil eder.

64 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 ... 0"

Tanımlı: MQMessage sınıfı

Veri Tipi: 32 ASCII karakteri simgeleyen 64 onaltılı karakterden oluşan dize

Sözdizimi: almak için: *actokenh $= MQMessage*.**AccountingTokenHex**

Ayarlamak için: *MQMessage.***AccountingTokenHex** = *actokenh $*

AccountingToken özelliği yerine AccountingTokenHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için “İleti Tanımlayıcısı özellikleri” sayfa 993 konusuna bakın.

ApplicationIdData özelliği

Okuma-yazma. MQMD ApplIdentityData-Message Identity Bağlamı 'nın bir parçası.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 32 karakterlik dize

Sözdizimi: almak için: *applid $= MQMessage*.**ApplicationIdData**

Ayarlamak için: *MQMessage*.**ApplicationIdData** = *applid $*

ApplicationOriginVeri özelliği

Okuma-yazma. MQMD ApplOriginVeri kısmı, ileti kaynağı bağlamının bir parçası.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 4 karakter dizgisi

Sözdizimi: almak için: *applor $= MQMessage*.**ApplicationOriginVerileri**

Ayarlamak için: *MQMessage*.**ApplicationOriginData** = *applor $*

BackoutCount özelliği

Salt okunur. MQMD BackoutCount.

Başlangıç değeri 0 olur.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *backoutct & = MQMessage***.BackoutCount**

CharacterSet özelliği

Okuma-yazma. MQMD CodedCharSetId.

Its initial value is the special value **MQCCSI_Q_MGR** .

CharacterSet seçeneği **MQCCSI_Q_MGR**olarak ayarlandıysa, WriteString yönteminde, yürürlükteki ülke değerine ilişkin kod sayfası karakter dönüştürmesi için kullanılır. Sunucu uygulamaları için, kullanılan kod sayfası kuyruk yöneticisinin kod sayfasıdır. İstemci uygulamaları için, varsayılan yürürlükteki ülke değeri kod sayfasıdır.

Örneğin:

```
msg.CharacterSet = MQCCSI_Q_MGR
msg.WriteString(chr$(n))
```

burada ' n', sıfırdan büyük ya da sıfıra eşit ve 255 'e eşit ya da daha küçük, arabelleğe yazılacak tek bir değer byte 'ında sonuçlar elde edilir.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: Almak İçin: *:30ccid& = MQMessage* **.CharacterSet**

Ayarlamak için: *MQMessage***.CharacterSet**= *ccid &*

Örnek

Kod sayfası 437 'de yazılan dizginin yazılmasını istiyorsanız, sorun:

```
Message.CharacterSet = 437
Message.WriteString ("string to be written")
```

Set the value you want in the CharacterSet before issuing any WriteString calls.

CorrelationId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQMD ' ye dahil edilmesi için CorrelationId . Ayrıca, kuyruktan ileti alınırken eşleştirilecek tanıtıcı.

Başlangıç değeri boş değerli.

Tanımlı: MQMessage sınıfı

Veri Tipi: 24 karakter dizgisi

Sözdizimi: almak için: *correlid $= MQMessage***.CorrelationId** : *MQMessage***.CorrelationId** = *correlid $*.

CorrelationId özelliği yerine CorrelationIdHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için “İleti Tanımlayıcısı özellikleri” sayfa 993 konusuna bakın.

CorrelationIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQMD ' ye dahil edilmesi için CorrelationId . Ayrıca, bir kuyruktan ileti alınırken eşleştirilecek CorrelationId ' nin de eşleşmesi gerekir.

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çifti tek karakteri "B", vb. temsil eder.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 ... 0".

Tanımlı: MQMessage sınıfı

Veri Tipi: 24 ASCII karakteri temsil eden 48 onaltılı karakterden oluşan dize

Sözdizimi: almak için: *correlidh $= MQMessage*.**CorrelationIdHex**

Ayarlamak için: *MQMessage*.**CorrelationIdHex** = *correlidh $*

CorrelationId özelliği yerine CorrelationIdHex 'i kullanmanız gerektiği zaman tartışması için bkz. “İleti Tanımlayıcısı özellikleri” sayfa 993 .

Kodlama özelliği

Okuma-yazma. Uygulama iletisi verilerinde sayısal değerler için kullanılan gösterimi tanımlayan MQMD alanı.

Başlangıç değeri, platforma göre değişen özel değer MQENC_NATIVE değeridir.

Bu özellik aşağıdaki yöntemler tarafından kullanılır:

- ReadDecimal2 yöntemi
- ReadDecimal4 yöntemi
- ReadDouble yöntemi
- ReadDouble4 yöntemi
- ReadFloat yöntemi
- ReadInt2 yöntemi
- ReadInt4 yöntemi
- ReadLong yöntemi
- ReadShort yöntemi
- ReadUInt2 yöntemi
- WriteDecimal2 yöntemi
- WriteDecimal4 yöntemi
- WriteDouble yöntemi
- WriteDouble4 yöntemi
- WriteFloat yöntemi
- WriteInt2 yöntemi
- WriteInt4 yöntemi
- WriteLong yöntemi
- WriteShort yöntemi
- WriteUInt2 yöntemi

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *encoding & = MQMessage*.**Kodlama** : *MQMessage***öğesini ayarlayın.Kodlama** = *encoding &*

İleti arabelleğiyle veri yazmaya hazırlanıyorsanız, alan kuyruk yöneticisi kendi veri dönüştürmesini gerçekleştiremeyecekse, bu alanı alan kuyruk yöneticisi altyapısının özellikleriyle eşleşecek şekilde ayarlamalısınız.

Süre bitimi özelliği

Okuma-yazma. MQMD süre bitimi zaman alanı, saniyenin onda biri olarak beklenir.

Başlangıç değeri, MQE_UNSıNISıT özel değeridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *expiry & = MQMessage*.**Süre Bitimi**

Ayarlamak için: *MQMessage*.**Süre Bitimi** = *süre bitimi ve*

Feedback özelliği

Okuma-yazma. MQMD geribildirim alanı.

İlk değeri, MQFB_NONE özel değeri.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. Geribildirim.

Sözdizimi: almak için: *feedback & = MQMessage*.**Geribildirim**

Ayarlamak için: *MQMessage*.**Geribildirim** = *geri bildirim ve*

Biçim özelliği

Okuma-yazma. MQMD biçimi alanı. İleti verilerinin niteliyi tanımlayan yerleşik ya da kullanıcı tanımlı bir biçemin adını verir.

İlk değeri, MQFMT_NONE özel değeridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 8 karakter dizgisi

Sözdizimi: almak için: *format $= MQMessage*.**Biçim**

Ayarlamak için: *MQMessage*.**Biçim** = *biçim $*

GroupId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için GroupId . Ayrıca, kuyruktan ileti alınırken eşleştirilecek tanıtıcı. Başlangıç değeri tüm boş değerlerde olur.

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
24 karakter dizgisi

Sözdizimi: almak için: *groupid $= MQMessage*.**GroupId**

Ayarlamak için: *MQMessage*.**GroupId** = *groupid $*

GroupId özelliği yerine GroupIdHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için bkz. “İleti Tanımlayıcısı özellikleri” sayfa 993 .

GroupIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için GroupId . Ayrıca, kuyruktan ileti alınırken eşleştirilecek tanıtıcı.

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çiftini temsil eder ve tek karakteri "B" vb. gösterir.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 ... 0".

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
24 ASCII karakteri simgeleyen 48 onaltılı karakterden oluşan dizgi.

Sözdizimi: almak için: *groupidh $= MQMessage*.**GroupIdHex**

Ayarlamak için: *MQMessage*.**GroupIdHex** = *groupidh $*

GroupId özelliği yerine GroupIdHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için bkz. “İleti Tanımlayıcısı özellikleri” sayfa 993 .

MessageData özelliği

Okuma-yazma. Bir iletinin tüm içeriğini karakter dizgisi olarak alır ya da ayarlar.

Tanımlı: MQMessage sınıfı

Veri Tipi: Varyant

Not: Bu özellik tarafından kullanılan veri tipi Variant, ancak MQAX bunun bir dizgi çeşitleme tipi olmasını bekliyor. Bu tipten başka bir çeşitleme geçerseniz, MQRC_OBJECT_TYPE_ERROR hatası döndürülür.

Sözdizimi: Almak Için: *String$ = MQMessage*.**MessageData**

Ayarlamak için: *MQMessage*.**MessageData** = *String$*

MessageFlags özelliği

Okuma yazma. Kesimlere ayırma denetim bilgileri belirten ileti işaretleri. Başlangıç değeri 0 olur.

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. MsgFlags (MQHOT).

Sözdizimi: almak için: *messageflags & = MQMessage*.**MessageFlags**

Ayarlamak için: *MQMessage*.**MessageFlags** = *messageflags &*

MessageId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQMD ' ye dahil edilmesi için MessageId . Ayrıca, kuyruktan ileti alınırken eşleştirilecek tanıtıcı.

Başlangıç değeri tüm boş değerlerde olur.

Tanımlı: MQMessage sınıfı

Veri Tipi: 24 karakter dizgisi

Sözdizimi: almak için: *messageid $= MQMessage*.**MessageId**

Ayarlamak için: *MQMessage*.**MessageId** = *messageid $*

MessageId özelliği yerine MessageIdHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için “İleti Tanımlayıcısı özellikleri” sayfa 993 konusuna bakın.

MessageIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQMD ' ye dahil edilmesi için MessageId . Ayrıca, bir kuyruktan ileti alınırken eşleştirilecek MessageId ' dir.

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çifti tek karakteri "B", vb. temsil eder.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 ... 0".

Tanımlı: MQMessage sınıfı

Veri Tipi: 24 ASCII karakteri temsil eden 48 onaltılı karakterden oluşan dize

Sözdizimi: almak için: *messageidh $= MQMessage*.**MessageIdHex**

Ayarlamak için: *MQMessage*.**MessageIdHex** = *messageidh $*

MessageId özelliği yerine MessageIdHex 'i kullanmanız gerektiği zaman hakkında daha fazla bilgi için “İleti Tanımlayıcısı özellikleri” sayfa 993 konusuna bakın.

MessageSequenceSayı özelliği

Okuma yazma. Grup içindeki bir iletiyi tanımlayan sıra bilgileri. Başlangıç değeri 1 'dir.

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. MsgSeqNumber (MQUZE).

Sözdizimi: almak için: *sequencenumber & = MQMessage*.**SequenceNumber**

Ayarlamak için: *MQMessage*.**SequenceNumber** = *sequencenumber &*

MessageType özelliği

Okuma-yazma. MQMD MsgType alanı.

İlk değeri MQMT_DATAGRAM ' tır.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. MsgType (MQUZE).

Sözdizimi: almak için: *msgtype & = MQMessage*.**MessageType**

Ayarlamak için: *MQMessage*.**MessageType** = *msgtype &*

Göreli konum özelliği

Okuma yazma. Bölümlenmiş bir iletteki göreli konum. Başlangıç değeri 0 olur.

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. Göreli Konum (MQUZE).

Sözdizimi: almak için: *offset & = MQMessage*.**Offset**

Ayarlamak için: *MQMessage*.**Offset** = *offset &*

OriginalLength özelliği

Okuma yazma. Kesimlere ayrılmış bir iletinin özgün uzunluğu. İlk değer MQOL_UNDEFINED değeridir.

Tanımlandığı yer:
MQMessage sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. OriginalLength (MQlong).

Sözdizimi: Almak Için: *originallength & = MQMessage*.**OriginalLength**

Ayarlamak için: *MQMessage*.**OriginalLength** = *originallength &*

Kalıcılık özelliği

Okuma-yazma. İletinin devamlılığı ayarı.

Başlangıç değeri MQPER_PERSISTENCE_AS_Q_DEF 'dir.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *persist & = MQMessage*.**Kalıcılık**

Ayarlamak için: *MQMessage*.**Kalıcılık** = *kalıcı saklama &*

Öncelik özelliği

Okuma-yazma. Mesajın önceliği var.

Başlangıç değeri, MQPRIO_PRIORIY_AS_Q_DEF özel değeridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *priority & = MQMessage*.**Öncelik**

Ayarlamak için: *MQMessage*.**Öncelik** = *priority &*

PutApplicationAd özelliği

Okuma-yazma. İleti kökeni bağlamının MQMD PutApplAdı-kısmı.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 28 karakterden oluşan dize

Sözdizimi: Almak Için: *putapplnm $= MQMessage*.**PutApplicationAdı**

Ayarlamak için: *MQMessage*.**PutApplicationName** = *putapplnm $*

PutApplicationTip özelliği

Okuma-yazma. İleti kökeni bağlamının MQMD PutAppltipi-kısmı.

Başlangıç değeri MQAT_NO_CONTEXT ' dir

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. PutApplTip (MQUZE).

Sözdizimi: Almak Için: *putappltp & = MQMessage*.**PutApplicationType**

Ayarlamak için: *MQMessage*.**PutApplicationType** = *putappltp &*

PutDateSaat özelliği

Okuma/yazma. Bu özellik, MQMD PutDate ve PutTime alanlarını birleştirir. Bunlar ileti kökeni bağlamının, iletinin ne zaman konduğunu gösteren kısımlarıdır.

ActiveX Extension, ActiveX tarih/saat biçimi ile WebSphere MQ MQMD ' de kullanılan Tarih ve Saat biçimleri arasında dönüştürür. Geçersiz bir PutDate ya da PutTime(PutTime) içeren bir ileti alındıysa, get yöntemi EMPTY olarak ayarlandıktan sonra PutDateTime özelliği.

İlk değeri BOŞ (EMPTY).

Tanımlı: MQMessage sınıfı

Veri Tipi: Tip 7 (tarih/saat) ya da EMPTY tipinde.

Sözdizimi: almak için: *datetime = MQMessage*.**PutDateTime**

Ayarlamak için: *MQMessage*.**PutDateTime** = *datetime*

ReplyToQueueManagerAd özelliği

Okuma-yazma. MQMD ReplyToQMgr alanı.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *replytoqmgr $= MQMessage*.**ReplyToQueueManagerName**

Ayarlamak için: *MQMessage*.**ReplyToQueueManagerName** = *replytoqmgr $*

ReplyToQueueName özelliği

Okuma-yazma. MQMD ReplyToQ alanı.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *replytoq $= MQMessage*.**ReplyToQueueName**

Ayarlamak için: *MQMessage*.**ReplyToQueueName** = *replytoq $*

Rapor özelliği

Okuma-yazma. İletinin Rapor seçenekleri.

İlk değeri MQRO_NONE olur.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. Rapor.

Sözdizimi: almak için: *report & = MQMessage*.**Rapor**

Ayarlamak için: *MQMessage*.**Rapor** = *rapor ve*

TotalMessageUzunluk özelliği

Salt okunur. MQGET tarafından alınan son iletinin uzunluğunu alır. İleti kısaltılmamışsa, bu değer MessageLength özelliğinin değerine eşittir.

Tanımlı: MQMessage sınıfı

Veri Tipi: Uzun

Sözdizimi: almak için: *totalmessabelength & = MQMessage*.**TotalMessageLength**

UserId özelliği

Okuma-yazma. MQMD UserIdentifier -Message Identity Context (İleti Tanıtıcısı Bağlamı) kısmı.

Başlangıç değeri boşluk karakteridir.

Tanımlı: MQMessage sınıfı

Veri Tipi: 12 karakter dizgisi

Sözdizimi: almak için: *userid $= MQMessage*.**UserId**

Ayarlamak için: *MQMessage*.**UserId** = *userid $*

ClearErrorCodes yöntemi

CompletionCode 'u MQCC_OK ve ReasonCode ' yi hem MQMessage sınıfı, hem de MQSession sınıfı için MQRC_NONE olarak sıfırlar.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**ClearErrorCodes ()**

ClearMessage yöntemi

Bu y "ntem, MQMessage nesnesinin veri arabelleğü b" lmesini temizler. Veri arabelleğindeki herhangi bir ileti verisi kaybedilir; çünkü MessageLength, DataLengthve DataOffset her şey sıfır olarak ayarlanır.

Message Descriptor (MQMD) kısmı etkilenmez; MQMessage nesnesini yeniden kullanmadan önce bir uygulamanın MQMD alanlarının bazılarını değiştirmesi gerekebilir. MQMD alanlarını ayarlamak için nesneyi yeni bir eşgörünüme sahip yeni bir eşgörünüme yenisiyle değiştirmek için Yeni düğmesini kullanın.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**ClearMessage()**

Okuma yöntemi

İleti arabelleğinden bayt dizisine kadar bir bayt dizisi okur. DataOffset artırılır ve Veri Uzunluğu okunan bayt sayısı kadar azaltılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Data = MQMessage.**Read**(len &)

Parametreler:
len &: Uzun. Okunabilmek için bayt cinsinden veri uzunluğu.

ReadBoolean yöntemi

İleti arabelleğindeki yürürlükteki konumdan 1 byte 'lık bir Boole değeri okur ve 2 baytlık bir Boole TRUE (-1) /YANLIŞ (0) değerini döndürür. DataOffset , bir birim tarafından artırılır ve Veri Uzunluğu bir birim tarafından azaltılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: *value = MQMessage*.**ReadBoolean**

ReadByte yöntemi

Bu yöntem, İleti Verileri arabelleğinden 1 byte 'ı okur ve DataOffset tarafından gönderme yapılan karakterle başlayarak, -128-127 aralığında bir Tamsayı (imli 2-bayt) tamsayı değeri olarak döndürür.

The method fails if MQMessage.DataLength is less than 1 when it is issued.

Yöntem başarılı olursa,DataOffset , 1 artırılır ve DataLength değeri 1 ile azaltılır. Yöntem başarılı olursa, 1 değerini artırılır.

İleti verilerinin byte 'ı imzalı bir ikili tamsayı olduğu varsayılır.

Tanımlı: MQMessage sınıfı

Sözdizimi: *integerv% = MQMessage*.**ReadByte**

ReadDecimal2 yöntemi

2 baytlık paketlenmiş onlu sayıyı okur ve imzalanmış 2 baytlık bir tamsayı değeri olarak döndürür. DataOffset , iki birim artırılır ve Veri Uzunluğu iki tarafından azaltılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: *value% = MQMessage*.**ReadDecimal2**

ReadDecimal4 yöntemi

4 baytlık paketlenmiş bir ondalık sayıyı okur ve imzalanmış 4 baytlık bir tamsayı değeri olarak döndürür. DataOffset , dört artırılır ve Veri Uzunluğu dört tarafından azaltılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *value & = MQMessage*.**ReadDecimal4**

ReadDouble yöntemi

This method reads 8 bytes from the Message Data buffer, starting with the byte referred to by DataOffset and returns it as a Double (signed 8-byte) floating point value.

The method fails if MQMessage.DataLength is less than 8 when it is issued.

Yöntem başarılı olursa,DataOffset , 8 artırılır ve DataLength değeri 8 tarafından azaltılır.

İleti verilerinin 8 karakterinin ikili kayar noktalı sayı olduğu varsayılır. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir. System/360 biçiminden dönüştürmenin desteklenmediğini unutmayın.

Tanımlı: MQMessage sınıfı

Sözdizimi: *doublev# = MQMessage*.**ReadDouble**

ReadDouble4 yöntemi

ReadDouble4 ve WriteDouble4 yöntemleri, ReadFloat ve WriteFloatyöntemlerine alternatiflerdir. Bunun nedeni, 4 baytlık IEEE kayar noktalı biçim biçimine dönüştürülebilmek için çok büyük olan 4 baytlık System/390 kayan noktalı ileti değerlerini destekledikleri için.

This method reads 4 bytes from the Message Data buffer, starting with the byte referred to by DataOffset and returns it as a Double (signed 8-byte) floating point value.

MQMessage.DataLength komutu verildiğinde 4 değerinden küçükse yöntem başarısız olur.

Yöntem başarılı olursa,DataOffset , 4 artırılır ve DataLength değeri 4 tarafından azaltılır.

İleti verilerinin 4 karakterinin bir ikili kayar noktalı sayı olduğu varsayılır. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir. System/360 biçiminden dönüştürmenin desteklenmediğini unutmayın.

Tanımlı: MQMessage sınıfı

Sözdizimi: *doublev# = MQMessage.***ReadDouble4**

ReadFloat yöntemi

This method reads 4 bytes from the Message Data buffer, starting with the byte referred to by DataOffset and returns it as a Single (signed 4-byte) floating point value.

MQMessage.DataLength komutu verildiğinde 4 değerinden küçükse yöntem başarısız olur.

Yöntem başarılı olursa,DataOffset , 4 artırılır ve DataLength değeri 4 tarafından azaltılır.

İleti verilerinin 4 karakterinin kayan noktalı bir sayı olduğu varsayılır. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir. System/360 biçiminden dönüştürmenin desteklenmediğini unutmayın.

Tanımlı: MQMessage sınıfı

Sözdizimi: *singlev! = MQMessage*.**ReadFloat**

ReadInt2 yöntemi

Yöntem, ReadShort yöntemi ile aynıdır.

Sözdizimi: *integerv% = MQMessage*.**ReadInt2**

ReadInt4 yöntemi

Bu yöntem ReadLong yöntemi ile aynıdır.

Sözdizimi: *bigint & = MQMessage*.**ReadInt4**

ReadLong yöntemi

This method reads 4 bytes from the Message Data buffer, starting with the byte referred to by DataOffset and returns it as a Long (signed 4-byte) integer value.

MQMessage.DataLength komutu verildiğinde 4 değerinden küçükse yöntem başarısız olur.

Yöntem başarılı olursa,DataOffset , 4 artırılır ve DataLength değeri 4 tarafından azaltılır.

İleti verilerinin 4 karakterinin ikili bir tamsayı olduğu varsayılır. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir.

Tanımlı: MQMessage sınıfı

Sözdizimi: *bigint & = MQMessage*.**ReadLong**

ReadNullTerminatedString yöntemi

Bu yöntem, dizgi gömülü boş karakterler içerebiliyorsa, ReadString yerine kullanılır.

This method reads the specified number of bytes from the message data buffer starting with the byte referred to by DataOffset and returns it as an ActiveX string. Dizilim, sondan önce gömülü bir boş değer içeriyorsa, döndürülen dizginin uzunluğu, boş değer olmadan önce yalnızca bu karakterleri yansıtacak şekilde azaltılır.

DataOffset is incremented and DataLength is decremented by the value specified regardless of whether the string contains embedded null characters.

The characters in the message data are assumed to be a string in the code page that is specified by the MQMessage.CharacterSet property. Uygulama için ActiveX gösterimine dönüştürme gerçekleştirilir.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: *string $ = MQMessage*.**ReadNullTerminatedString(uzunluk &)**

Parametreler:
uzunluk & *Uzun.* Bayt cinsinden dizgi alanı uzunluğu.

ReadShort yöntemi

Bu yöntem, İleti Verileri arabelleğinden 2 byte 'ı okur ve DataOffset tarafından adlandırılan bayt ile başlayarak, onu Tamsayı (imli 2 bayt) değeri olarak döndürür.

The method fails if MQMessage.DataLength is less than 2 when it is issued.

Yöntem başarılı olursa,DataOffset , 2 artırılır ve DataLength değeri 2 tarafından azaltılır.

İleti verilerinin 2 karakterinin ikili bir tamsayı olduğu varsayılır. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir.

Tanımlı: MQMessage sınıfı

Sözdizimi: *integerv% = MQMessage*.**ReadShort**

ReadString yöntemi

This method reads n bytes from the Message Data buffer starting with the byte referred to by DataOffset and returns it as an ActiveX string.

MQMessage.DataLength komutu verildiğinde n değerinden küçükse yöntem başarısız olur.

Yöntem başarılı olursa,DataOffset , n artırılır ve DataLength değeri n tarafından azaltılır.

The n characters of message data are assumed to be a string in the code page that is specified by the MQMessage.CharacterSet property. Uygulama için ActiveX gösterimine dönüştürme gerçekleştirilir.

Tanımlı: MQMessage sınıfı

Sözdizimi: *stringv $= MQMessage*.**ReadString***(uzunluk &)*

Değiştirge
uzunluk ve Uzun. Bayt cinsinden dizgi alanı uzunluğu.

ReadUInt2 yöntemi

Bu yöntem, İleti Verileri arabelleğinden 2 byte 'ı okur ve DataOffset tarafından adlandırılan bayt ile başlayarak, onu uzun (4 baytlık imzalı) bir tamsayı değeri olarak döndürür.

The method fails if MQMessage.DataLength is less than 2 when it is issued.

Yöntem başarılı olursa,DataOffset , 2 artırılır ve DataLength değeri 2 tarafından azaltılır.

İleti verilerinin 2 baytı işaretsiz bir ikili tamsayı olarak kabul edilir. Kodlama, MQMessage.Encoding özelliği tarafından belirtilir.

Tanımlı: MQMessage sınıfı

Sözdizimi: *bigint & = MQMessage*.**ReadUInt2**

ReadUnsignedByte yöntemi

Bu yöntem, İleti Verileri arabelleğinden 1 byte 'ı okur ve DataOffset tarafından belirtilen bayt ile başlayarak, 0-255 aralığında bir Tamsayı (imli 2-bayt) tamsayı değeri olarak döndürür.

The method fails if MQMessage.DataLength is less than 1 when it is issued.

Yöntem başarılı olursa,DataOffset , 1 artırılır ve DataLength değeri 1 ile azaltılır. Yöntem başarılı olursa, 1 değerini artırılır.

İleti verilerinin 1 karakterinin imzalanmamış bir ikili tamsayı olduğu varsayılır.

Tanımlı: MQMessage sınıfı

Sözdizimi: *integerv% = MQMessage*.**ReadUnsignedByte**

ReadUTF yöntemi

This method reads a UTF format string from the message starting with the byte referred to by DataOffset and returns it as an ActiveX string. İletideki dizgi, 2 byte 'lık bir uzunluktan ve ardından karakter verilerinin izlediği bir dizilim içerir.

MQMessage.DataLength komutu verildiğinde dizgi uzunluğundan az olduğunda yöntem başarısız olur.

DataOffset is incremented by the string length and DataLength is decremented by the string length if the method succeeds.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: *value $= MQMessage*.**ReadUTF**

ResizeBuffer yöntemi

Bu yöntem, İleti Verileri arabelleğini tutmak için dahili olarak ayrılmış olan depolama miktarını değiştirir. Uygulama, otomatik arabellek yönetimi üzerinde bazı denetim sağlar. Bu durumda, uygulama büyük bir iletiyle başa çıkabileceğini biliyorsa, yeterli büyüklükte bir arabelleğin ayrılmasını sağlayabilirler. Uygulamanın bu çağrıyı kullanması gerekmez; yoksa, otomatik arabellek yönetim kodu arabellek büyüklüğünü sığacak şekilde büyütür.

Arabelleğin büyüklüğünü, yürürlükteki MessageLengthdeğerinden küçük olacak şekilde yeniden boyutlandırırsanız, veri kaybı riskine girmenize neden olur. Veri kaybederse, yöntem MQCC_UYARY için bir CompletionCode ve MQRC_DATA_TRUNCATED için ReasonCode değerini döndürür.

Arabelleğin büyüklüğünü, **DataOffset** özelliğinin değerinden daha küçük olacak şekilde yeniden boyutlandırırsanız:

- **DataOffset** özelliği, yeni arabelleğin sonuna işaret edecek şekilde değiştirildi
- **DataLength** özelliği sıfır olarak ayarlandı
- **MessageLength** özelliği yeni arabellek büyüklüğü olarak değiştirildi

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: *MQMessage*.**ResizeBuffer**(*Uzunluk &*)

Parametre:
Uzunluk ve Uzun. Karakterlerde gerekli boyut.

Yazma yöntemi

Veri Göreli Konumu ile gönderme yapılan konumdaki bayt dizisinden ileti arabelleğine bir bayt dizisi yazar. Gerekirse, arabelleğin uzunluğu (MQMessage.MQMessageLength), bayt dizisinin tam uzunluğuna sığması için genişletilir. DataOffset , yöntem başarılı olursa yazılan bayt sayısı kadar artırılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**Write**(değer)

Parametreler:
data: bir bayt dizisi ya da bir bayt dizisine ilişkin değişken başvurusu

WriteBoolean yöntemi

İleti arabelleğindeki, 2 baytlık Boole değerinden 1 byte 'lık bir Boole değerini yazar. DataOffset bir artırılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteBoolean**(değer)

Parametre:
değer: Boole (2-bayt). Yazılacak değer.

WriteByte yöntemi

Bu yöntem, imzalanmış 2 baytlık bir tamsayı değerini alır ve bunu Message Data arabelleğiyle DataOffsettarafından belirtilen konumda 1 byte 'lık ikili sayı olarak yazar. Önceden arabelleğindeki konumdaki verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset bir artırılır.

Belirtilen değer -128-127 aralığında olmalıdır. Doğru değilse, yöntem CompletionCode MQCC_FAILED ve ReasonCode MQRC_WRITE_VALUE_ERROR ile döner.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteByte**(*value%*)

Parametre: *value%* Tamsayı. Yazılacak değer.

WriteDecimal2 yöntemi

2 baytlık bir paketlenmiş onlu sayı olarak imzalı 2 baytlık bir tamsayı yazar. DataOffset , iki tarafından artırılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteDecimal2**(değer%)

Parametre:
değer% Tamsayı. Yazılacak değer.

WriteDecimal4 yöntemi

4 baytlık paketlenmiş bir ondalık sayı olarak, imzalı 4 baytlık bir tamsayıyı yazar. DataOffset , dört tarafından artırılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WritedDecimal4**(değer &)

Parametre:
değer ve Uzun. Yazılacak değer.

WriteDouble yöntemi

Bu yöntem, imzalanmış 8 baytlık bir kayan noktalı değer alır ve bunu İleti Verileri arabelleğiyle DataOffsettarafından adlandırılan konumdan başlayarak 8 baytlık bir kayan noktalı sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset 8 artırılır.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen kayan nokta gösterimine dönüştürür. *System/360 biçimine dönüştürme desteklenmiyor.*

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteDouble**(*value#*)

Parametre:
değer# Çift. Yazılacak değer.

WriteDouble4 yöntemi

See "ReadDouble4 yöntemi" sayfa 1042 for a description of when ReadDouble4 and WriteDouble4 should be used in place of ReadFloat and WriteFloat.

Bu yöntem, imzalanmış 8 baytlık bir kayan noktalı değer alır ve bunu İleti Verileri arabelleğiyle DataOffsettarafından adlandırılan konumdan başlayarak 4 baytlık bir kayan sayı olarak yazar.

Yöntem başarılı olursa,DataOffset , 4 artırılır.

Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen kayan nokta gösterimine dönüştürür. *System/360 biçimine dönüştürme desteklenmiyor.*

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteDouble4 (değer#)**

Parametre: *value#* Çift. Yazılacak değer.

WriteFloat yöntemi

Bu yöntem, imzalanmış 4 baytlık bir kayan noktalı değer alır ve bunu Message Data arabelleğinine, DataOffsettarafından başvurulan karakterde başlayan 4 baytlık bir kayan noktalı sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset , 4 artırılır.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen ikili gösterimine dönüştürür. *System/360 biçimine dönüştürme desteklenmiyor.*

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage***.WriteFloat**(*value!*)

Parametre *değer!* Yüzmek. Yazılacak değer.

WriteInt2 yöntemi

Bu yöntem WriteShort yöntemi ile aynıdır.

Sözdizimi: Call *MQMessage***.WriteInt2**(*değer%*)

Parametre *değer%* Tamsayı. Yazılacak değer.

WriteInt4 yöntemi

Bu yöntem WriteLong yöntemi ile aynıdır.

Sözdizimi: Call *MQMessage***.WriteInt4**(*değer &*)

Parametre *değer ve* Uzun. Yazılacak değer.

WriteLong yöntemi

Bu yöntem, imzalanmış 4 baytlık bir tamsayı değerini alır ve Message Data arabelleğiyle DataOffsettarafından adlandırılan bayt 'dan başlayarak 4 baytlık ikili bir sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset , 4 artırılır.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen ikili gösterimine dönüştürür.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage***.WriteLong**(*value &*)

Parametre *değer ve* Uzun. Yazılacak değer.

WriteNullTerminatedString yöntemi

Bu yöntem, olağan bir WriteString gerçekleştirir ve kalan tüm baytları boş değerle belirtilen uzunluğa kadar destekler. İlk yazma dizgisi tarafından yazılan baytların sayısı belirtilen uzunluğa eşitse, boş değer yazılanmaz. Byte sayısı belirtilen uzunluğu aşarsa bir hata oluştu (neden kodu MQRC_WRITE_VALUE_ERROR) ayarlanır.

Yöntem başarılı olursa,DataOffset belirtilen uzunluğa göre artırılır.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteNullTerminatedString***(value$, length &)*

Parametreler:
$String değeri. Yazılacak değer.

uzunluk ve Uzun. Bayt cinsinden dizgi alanı uzunluğu.

WriteShort yöntemi

Bu yöntem, imzalanmış 2 baytlık bir tamsayı değerini alır ve Message Data arabelleğiyle DataOffsettarafından adlandırılan bayt 'dan başlayarak 2 baytlık ikili bir sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) uzatır.

Yöntem başarılı olursa,DataOffset , 2 artırılır.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen ikili gösterimine dönüştürür.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteShort**(*value%*)

Parametre *değer%* Tamsayı. Yazılacak değer.

WriteString yöntemi

This method takes an ActiveX string and writes it into the Message Data buffer starting at the byte referred to by DataOffset. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) uzatır.

Yöntem başarılı olursa,DataOffset , bayt cinsinden dizgi uzunluğuna göre artırılır.

Yöntem, karakterleri MQMessage.CharacterSet özelliği tarafından belirtilen kod sayfasına dönüştürür.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteString**(*value $*)

Parametre *değer $* Dize. Yazılacak değer.

WriteUInt2 yöntemi

Bu yöntem, imzalanmış 4 baytlık bir tamsayı değerini alır ve Message Data arabelleğiine, DataOffsettarafından gönderme yapılan bayttan başlayarak 2 baytlık imzalanmamış bir ikili sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset , 2 artırılır.

Yöntem, MQMessage.Encoding özelliği tarafından belirtilen ikili gösterimine dönüştürür. Belirlenen değer 0-2 * *16-1 aralığında olmalıdır. Bu yöntem, CompletionCode MQCC_FAILED ve ReasonCode MQRC_WRITE_VALUE_ERROR ile döndüren yöntem değildir.

Tanımlı: MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteUInt2**(*değer &*)

Parametre *değer ve* Uzun. Yazılacak değer.

WriteUnsignedByte yöntemi

Bu yöntem, imzalanmış 2 baytlık bir tamsayı değerini alır ve Message Data arabelleğiine, DataOffsettarafından başvurulan karakterde başlayan 1 baytlık imzalanmamış bir ikili sayı olarak yazar. Arabelleğindeki bu konumlarda bulunan verilerin yerine geçer ve gerekirse arabelleğin uzunluğunu (MQMessage.MessageLength) genişletir.

Yöntem başarılı olursa,DataOffset , 1 artırılır.

Belirlenen değer, 0-255 aralığında olmalıdır. Bu yöntem, CompletionCode MQCC_FAILED ve ReasonCode MQRC_WRITE_VALUE_ERROR ile döndüren yöntem değildir.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteUnsignedByte**(*value%*)

Parametre *değer%* Tamsayı. Yazılacak değer.

WriteUTF yöntemi

Bu yöntem, bir ActiveX dizgisini alır ve UTF biçiminde geçerli konumda ileti veri arabelleğiyle yazar. Yazılan veriler, 2 baytlık karakter uzunluğunun ve ardından karakter verilerinin izlediği bir uzunluktan oluşur. Yöntem başarılı olursa,DataOffset , dizginin uzunluğuna göre artırılır.

Tanımlandığı yer:
MQMessage sınıfı

Sözdizimi: Call *MQMessage*.**WriteUTF**(value$)

Parametre:
değer $Dizgisi. Yazılacak değer.

MQPutMessageSeçenekleri sınıfı

Bu sınıf, bir WebSphere MQ Queue 'ye ileti koyma işlemini denetleyen çeşitli seçenekleri sarsalır.

Bulundurma

The MQPutMessageOptions class is contained by the MQSession class.

Yaratma

Yeni seçeneği, yeni bir MQPutMessageSeçenekleri nesnesi yaratır ve tüm özelliklerini ilk değerlerine ayarlar.

Diğer bir seçenek olarak, MQSession sınıfının AccessPutMessageOptions yöntemini kullanın.

Sözdizimi

Dim *pmo* **Farklı Yeni MQPutMessageSeçenekleri** ya da

Set *pmo* = **Yeni MQPutMessageSeçenekleri**

Özellikler

- "CompletionCode özelliği" sayfa 1049.
- "Seçenekler özelliği" sayfa 1050.
- "ReasonCode özelliği" sayfa 1050.
- "ReasonName özelliği" sayfa 1050.
- "RecordFields özelliği" sayfa 1050.
- "ResolvedQueueManagerName özelliği" sayfa 1051.
- "ResolvedQueueAd özelliği" sayfa 1051.

Yöntemler

- "ClearErrorCodes yöntemi" sayfa 1051.

CompletionCode özelliği

Salt okunur. Nesneye yönelik olarak verilen son yöntem ya da özellik erişimi tarafından ayarlanan tamamlanma kodunu döndürür.

Tanımlı: MQPutMessageSeçenekleri sınıfı

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *completioncode & = PutOpts*.**CompletionCode**

Seçenekler özelliği

Okuma-yazma. MQPMO Seçenekleri alanı. Bu alanın ilk değeri MQPMO_NONE olur. Ek bilgi için MQPMO seçenekleribaşlıklı konuya bakın.

Tanımlı: MQPutMessageSeçenekleri Sınıfı.

Veri Tipi: Uzun

Sözdizimi: almak için: *options & = PutOpts*.**Seçenekler**

Ayarlamak için: *PutOpts*.**Seçenekler** = *seçenekler ve*

MQPMO_PASS_IDENTITY_CONTEXT ve MQPMO_PASS_ALL_CONTEXT seçenekleri desteklenmiyor.

ReasonCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimiyle belirlenen neden kodunu döndürür.

Tanımlı: MQPutMessageSeçenekleri sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasoncode & = PutOpts*.**ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Tanımlı: MQPutMessageSeçenekleri sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= PutOpts*.**ReasonName**

RecordFields özelliği

Okuma-yazma. Dağıtım listesine bir ileti yerleştirilirken, kuyruklar başına hangi alanların uyarlanaca § ını belirten işaretler. Başlangıç değeri sıfır.

Bu özellik, MQI MQPMO yapısındaki PutMsgRecFields işaretlerine karşılık gelir. MQI 'de, bu işaretler, MQPUT tarafından hangi alanların (MQPMR yapısında) bulunduğunu ve kullanıldığını denetler. Bir MQPutMessageOptions nesnesinde, bu alanlar her zaman vardır ve işaretler, yalnızca put tarafından kullanılan alanları etkiler. Daha ayrıntılı bilgi için *WebSphere MQ Application Programming Reference* adlı kılavuza bakın.

Tanımlandığı yer:
MQPutMessageSeçenekleri sınıfı

Veri Tipi:
Uzun

Sözdizimi: almak için: *recordfields & = PutOpts*.**RecordFields**

Ayarlamak için: *PutOpts*.**RecordFields** = *recordfields &*

ResolvedQueueManagerName özelliği

Salt okunur. MQPMO ResolvedQMgrAd alanı. Ayrıntılar için ResolvedQMgrName (MQCHAR48) konusuna bakın. Başlangıç değeri boşluk karakteridir.

Tanımlı: MQPutMessageSeçenekleri sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *qmgr $= PutOpts*.**ResolvedQueueManagerName**

ResolvedQueueAd özelliği

Salt okunur. MQPMO ResolvedQName alanı. Ayrıntılı bilgi için ResolvedQName (MQCHAR48) başlıklı konuya bakın. Başlangıç değeri boşluk karakteridir.

Tanımlı: MQPutMessageSeçenekleri sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: almak için: *qname $= PutOpts*.**ResolvedQueueAdı**

ClearErrorCodes yöntemi

CompletionCode öğesini, hem MQPutMessageSeçenekleri sınıfı, hem de MQSession sınıfı için MQCC_NONE ve MQRC_NONE MQRC_NONE değerine döndürür.

Tanımlı: MQPutMessageSeçenekleri sınıfı

Sözdizimi: Call *PutOpts*.**ClearErrorCodes ()**

MQGetMessageSeçenekleri sınıfı

Bu sınıf, bir WebSphere MQ kuyruğundan ileti alma işlemini denetleyen çeşitli seçenekleri sarsalıyor.

Bulundurma

The MQGetMessageOptions class is contained by the MQSession class.

Yaratma

Yeni seçeneği, yeni bir MQGetMessageSeçenekleri nesnesi yaratır ve tüm özelliklerini ilk değerlerine ayarlar.

Diğer bir seçenek olarak, MQSession sınıfının AccessGetMessageOptions yöntemini kullanın.

Özellikler

- "CompletionCode özelliği" sayfa 1052
- "MatchOptions özelliği" sayfa 1052
- "Seçenekler özelliği" sayfa 1052
- "ReasonCode özelliği" sayfa 1052
- "ReasonName özelliği" sayfa 1052
- "ResolvedQueueAd özelliği" sayfa 1053
- "WaitInterval özelliği" sayfa 1053

Yöntemler

- "ClearErrorCodes yöntemi" sayfa 1053

Sözdizimi

Dim *gmo* **Yeni MQGetMessageSeçenekleri Olarak** ya da

Set *gmo* = **Yeni MQGetMessageSeçenekleri**

CompletionCode özelliği

Salt okunur. Nesneye yönelik olarak verilen son yöntem ya da özellik erişimi tarafından ayarlanan tamamlanma kodunu döndürür.

Tanımlı: MQGetMessageSeçenekleri Sınıfı.

Veri Tipi: Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: Alınmak için: *completioncode & = GetOpts*.**CompletionCode**

MatchOptions özelliği

Okuma-yazma. MQGET için kullanılan seçim ölçütlerini denetleyen seçenekler. Başlangıç değeri: MQMO_MATCH_MSG_ID + MQMO_MATCH_COREL_ID.

Tanımlandığı yer:
MQGetMessageSeçenekleri sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. MatchOptions (MQUZE).

Sözdizimi: Alınmak için: *matchoptions & = GetOpts*.**MatchOptions**

Ayarlamak için: *GetOpts*.**MatchOptions** =*matchoptions &*

Seçenekler özelliği

Okuma-yazma. MQGMO Seçenekleri alanı. Ayrıntılar için Seçenekler konusuna bakın. İlk değer MQGMO_NO_DUR değeridir.

Tanımlı: MQGetMessageSeçenekleri Sınıfı.

Veri Tipi: Uzun

Sözdizimi: Alınmak için: *options & = GetOpts*.**Seçenekler** ayarlamak için: *GetOpts*.**Seçenekler** = *seçenekler ve*

ReasonCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimiyle belirlenen neden kodunu döndürür.

Tanımlı: MQGetMessageSeçenekleri sınıfı

Veri Tipi: Uzun

Değerler:

- Bkz. API neden kodları.

Sözdizimi: Alınmak için: *reasoncode & = GetOpts*.**ReasonCode**

ReasonName özelliği

Salt okunur. En son neden kodunun simgesel adını döndürür. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".
Tanımlı: MQGetMessageSeçenekleri sınıfı

Veri Tipi: Dize

Değerler:

- Bkz. API neden kodları.

Sözdizimi: Alınmak için: *reasonname $= MQGetMessageOptions*.**ReasonName**

ResolvedQueueAd özelliği

Salt okunur. MQGMO ResolvedQName alanı. Ayrıntılı bilgi için ResolvedQName (MQCHAR48) başlıklı konuya bakın. Başlangıç değeri boşluk karakteridir.

Tanımlı: MQGetMessageSeçenekleri sınıfı

Veri Tipi: 48 karakterden oluşan dize

Sözdizimi: Alınmak için: *qname $= GetOpts*.**ResolvedQueueAdı**

WaitInterval özelliği

Okuma/yazma. MQGMO WaitInterval alanı. Options özelliği tarafından bekleme işlemi istendiyse, Get 'in uygun bir ileti gelmesi için bekleyeceği süre üst sınırı (milisaniye olarak). Bu alanın başlangıç değeri 0 (0). MQGMO seçeneklerine ilişkin ayrıntılar için bakınız: MQGMO.

Tanımlı: MQGetMessageSeçenekleri sınıfı

Veri Tipi: Uzun

Sözdizimi: Alınmak için: *wait & = GetOpts*.**WaitInterval**

Ayarlamak için: *GetOpts*.**WaitInterval** = wait &

ClearErrorCodes yöntemi

CompletionCode öğesini, hem MQGetMessageOptions sınıfı, hem de MQSession sınıfı için MQCC_NONE ve MQRC_NONE MQRC_NONE olarak sıfırlar.

Tanımlı: MQGetMessageSeçenekleri sınıfı

Sözdizimi: Call *GetOpts*.**ClearErrorCodes ()**

MQDistributionList sınıfı

Bu sınıf, çıkış için yerel, uzak ya da diğer ad derlemini sarsalıyor.

Yaratma

yeni , yeni bir MQDistributionList nesnesi yaratır.

Diğer bir seçenek olarak, MQQueueManager sınıfının AddDistributionList yöntemini kullanın.

Özellikler

- “AlternateUsertanıtıcı özelliği” sayfa 1054
- “CloseOptions özelliği” sayfa 1054
- “CompletionCode özelliği” sayfa 1054
- “ConnectionReference özelliği” sayfa 1055
- “FirstDistributionListItem özelliği” sayfa 1055
- “IsOpen özelliği” sayfa 1055
- “OpenOptions özelliği” sayfa 1055

- “ReasonCode özelliği” sayfa 1055
- “ReasonName özelliği” sayfa 1056

Yöntem

- “AddDistributionListItem yöntemi” sayfa 1056
- “ClearErrorCodes yöntemi” sayfa 1056
- “Yöntemi kapat” sayfa 1056
- “Open yöntemi” sayfa 1057
- “Put yöntemi” sayfa 1057

Sözdizimi

dim *distlist.***As New MQDistributionList** ya da **Set** *distlist* = **New MQDistributionList**

AlternateUsertanıtıcı özelliği

Okuma-yazma. Açıldığında kuyruklar listesine erişimi doğrulamak için kullanılan diğer kullanıcı kimliği.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
12 karakterden oluşan dizgi

Sözdizimi: almak için: *altuser $= MQDistributionList*.**AlternateUserTanıtıcısı**

Ayarlamak için: *MQDistributionList*.**AlternateUserId** = *altuser $*

CloseOptions özelliği

Okuma-yazma. Dağıtım listesi kapatıldığında ne olacağını denetlemek için kullanılan seçenekler. İlk değer MQCO_NONE olur.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCO_NONE
- MQCO_DELETE
- MQCO_DELETE_PURGE

Sözdizimi: almak için: *closeopt & = MQDistributionList*.**CloseOptions**

Ayarlamak için: *MQDistributionList*.**CloseOptions** = *closeopt &*

CompletionCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimi tarafından ayarlanan tamamlanma kodu.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı

- MQCC_FAILED

Sözdizimi: almak için: *completioncode & = MQDistributionList*.**CompletionCode**

ConnectionReference özelliği

Okuma-yazma. Dağıtım listesinin ait olduğu kuyruk yöneticisi.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
MQQueueManager

Sözdizimi: almak için: *set queuemanger = MQDistributionList*.**ConnectionReference**

Şu şekilde ayarlayın: *set MQDistributionList*. **ConnectionReference** = *queuemanager*

FirstDistributionListItem özelliği

Salt okunur. Dağıtım listesiyle ilişkili ilk dağıtım listesi öğesi nesnesi.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
MQDistributionListÖğesi

Değerler:

Sözdizimi: almak için: *set distributionlistitem = MQDistributionList*.**FirstDistributionListItem**

IsOpen özelliği

Salt okunur.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Boole

Değerler:

- TRUE (-1)
- FALSE (0)

Sözdizimi: almak için: *IsOpen = MQDistributionList*.**IsOpen**

OpenOptions özelliği

Okuma-yazma. Dağıtım listesi açıldığında kullanılacak seçenekler.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. MQPMO seçenekleri.

Sözdizimi: almak için: *openopt & = MQDistributionList*.**OpenOptions**

Ayarlamak için: *MQDistributionList.***OpenOptions** = openopt &

ReasonCode özelliği

Salt okunur. Nesneye karşı verilen son yöntem ya da özellik erişimi tarafından ayarlanan neden kodu.

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. API neden kodları.

Sözdizimi: almak için: *reasoncode & = MQDistributionList*.**ReasonCode**

ReasonName özelliği

Salt okunur. ReasonCodeiçin simgesel ad. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Tanımlandığı yer:
MQDistributionList sınıfı

Veri Tipi:
Dizgi

Değerler:
Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQDistributionList*.**ReasonName**

AddDistributionListItem yöntemi

Yeni bir MQDistributionListöğe nesnesi yaratır ve bunu dağıtım listesi nesnesiyle ilişkilendirir. Kuyruk adı parametresi zorunludur.

Dağıtım listesi öğesinin DistributionList özelliği iye dağıtım listesi olarak ayarlanır ve dağıtım listesinin FirstDistributionListItem özelliği bu yeni dağıtım listesi öğesine gönderme yapmak için ayarlanır.

Yeni dağıtım listesi öğesi için, PreviousDistributionListItem özelliği hiçbir şeye ayarlanmaz ve NextDistributionListItem özelliği önceden ilk olan herhangi bir dağıtım listesi öğesine gönderme yapmak için ayarlanır ya da daha önce hiç değilse (yani, zaten var olanların önüne yeni eklenen yeni eklenen) bir öğe için hiçbir şey ifade etmimez.

Bu, dağıtım listesi açıksa bir hata döndürür.

Tanımlandığı yer:
MQDistributionList sınıfı

Sözdizimi: set distributionlistitem = *MQDistributionList*.**AddDistributionListItem** (QName$, QMgrName$)

Parametreler:

QName$ Dize. WebSphere MQ kuyruğunun adı.

QMgrName$ Dize. WebSphere MQ kuyruk yöneticisinin adı.

ClearErrorCodes yöntemi

CompletionCode öğesini, hem MQDistributionList sınıfı, hem de MQSession sınıfı için MQCC_NONE ve MQRC_NONE MQRC_NONE olarak sıfırlar.

Tanımlandığı yer:
MQDistributionList sınıfı

Sözdizimi: Ara *MQDistributionList*.**ClearErrorCodes**()

Yöntemi kapat

Geçerli Kapanış seçeneklerinin geçerli değerini kullanarak bir dağıtım listesini kapatır.

Tanımlandığı yer:
MQDistributionList sınıfı

Sözdizimi: Ara *MQDistributionList*.**Kapat**()

Open yöntemi

AlternateUsertanıtıcısı ile yürürlükteki nesneyle ilişkili dağıtım listesi öğelerinin QueueManagerAd özelliklerinin (uygun olduğu) QueueName ve (uygun olduğu yerlerde) tarafından belirtilen kuyrukların her birini açar.

Tanımlandığı yer:
MQDistributionList sınıfı

Sözdizimi: Ara *MQDistributionList*.**Open**()

Put yöntemi

Dağıtım listesiyle ilişkili dağıtım listesi öğeleriyle tanımlanan kuyrukların her birine bir ileti yerleştirir.

Tanımlandığı yer:

MQDistributionList sınıfı

Sözdizimi

Call MQDistributionList.**Koy**(Message, PutMsgOptions&)

Parametreler

Konulacak iletiyi gösteren*İleti* MQMessage nesnesi.

Koyma işlemini denetleme seçeneklerini içeren*PutMsgSeçenekleri* MQPutMessageOptions nesnesi. Belirtilmemişse, varsayılan PutMessageSeçenekleri kullanılır.

Bu yöntem, bir MQMessage nesnesini değiştirge olarak alır. Aşağıdaki dağıtım listesi öğesi özellikleri, bu yöntemin bir sonucu olarak değiştirilebilir:

- CompletionCode
- ReasonCode
- ReasonName
- MessageId
- MessageIdOnaltılı
- CorrelationId
- CorrelationIdOnaltılı
- GroupId
- GroupIdOnaltılı
- Geribildirim
- AccountingToken
- AccountingTokenAltılı

MQDistributionListÖğe sınıfı

Bu sınıf, MQOR, MQRR ve MQPMR yapılarını sarsalıyor ve bunları sahip olan bir dağıtım listesiyle ilişkilendirir.

Yaratma

MQDistributionList sınıfının AddDistributionListItem Yöntemini kullanın.

Özellikler

Yöntemler

- “AccountingToken özelliği” sayfa 1059.
- “AccountingTokenHex özelliği” sayfa 1059.
- “CompletionCode özelliği” sayfa 1059.
- “CorrelationId özelliği” sayfa 1059.
- “CorrelationIdHex özelliği” sayfa 1060.
- “DistributionList özelliği” sayfa 1060.
- “Feedback özelliği” sayfa 1060.
- “GroupId özelliği” sayfa 1060.
- “GroupIdHex özelliği” sayfa 1060.
- “MessageId özelliği” sayfa 1061.
- “MessageIdHex özelliği” sayfa 1061.
- “NextDistributionListItem özelliği” sayfa 1061.
- “PreviousDistributionListItem özelliği” sayfa 1061.
- “QueueManagerAd özelliği” sayfa 1062.
- “QueueName özelliği” sayfa 1062.
- “ReasonCode özelliği” sayfa 1062.
- “ReasonName özelliği” sayfa 1062.
- “ClearErrorCodes yöntemi” sayfa 1062.

Özellikler

- AccountingToken özelliği
- AccountingTokenHex özelliği
- CompletionCode özelliği
- CorrelationId özelliği
- CorrelationIdHex özelliği
- DistributionList özelliği
- Feedback özelliği
- GroupId özelliği
- GroupIdHex özelliği
- MessageId özelliği
- MessageIdHex özelliği
- NextDistributionListItem özelliği
- PreviousDistributionListItem özelliği
- QueueManagerAd özelliği
- QueueName özelliği
- ReasonCode özelliği
- ReasonName özelliği

Yöntemler:

- ClearErrorCodes yöntemi

Yaratma:

MQDistributionList sınıfının AddDistributionListItem Yöntemini kullanın.

AccountingToken özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için AccountingToken . Başlangıç değeri tüm boş değerlerde olur.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
32 karakter dizgisi

Sözdizimi: almak için: *accountingtoken $= MQDistributionListöğesi*.**AccountingToken**

Ayarlamak için: *MQDistributionListItem*.**AccountingToken** = *accountingtoken $*

AccountingTokenHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için AccountingToken .

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çiftini temsil eder ve tek karakteri "B" vb. gösterir.

64 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 ... 0".

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
32 ASCII karakteri yeniden sunan, 64 onaltılı karakterden oluşan dizgi.

Sözdizimi: almak için: *accountingtokenh $= MQDistributionListItem*.**AccountingTokenHex**

Ayarlamak için: *MQDistributionListItem*.**AccountingTokenHex** = *accountingtokenh $*

CompletionCode özelliği

Salt okunur. Son açma ya da koyma isteği tarafından belirlenen tamamlanma kodu, sahip olan dağıtım listesi nesnesine ilişkin olarak verildi.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
Uzun

Değerler:

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *completioncode $= MQDistributionListItem*.**CompletionCode**

CorrelationId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için CorrelId (CorrelId). Başlangıç değeri tüm boş değerlerde olur.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 karakter dizgisi

Sözdizimi: almak için: *correlid $= MQDistributionListItem*.**CorrelationId**

Ayarlamak için: *MQDistributionListItem*.**CorrelationId** = *correlid $*

CorrelationIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için CorrelId (CorrelId).

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çiftini temsil eder ve tek karakteri "B" vb. gösterir.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 .. 0".

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 ASCII karakteri simgeleyen 48 onaltılı karakterden oluşan dizgi.

Sözdizimi: almak için: *correlidh $= MQDistributionListItem*.**CorrelationIdHex**

Ayarlamak için: *MQDistributionListItem*.**CorrelationIdHex** = *correlidh $*

DistributionList özelliği

Salt okunur. Bu dağıtım listesi öğesinin ilişkilendirildiği dağıtım listesi.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
MQDistributionList

Sözdizimi: almak için: *set distributionlist = MQDistributionListItem*.**DistributionList**

Feedback özelliği

Okuma-yazma. Bir ileti kuyruğuna konduğunda iletinin MQPMR ' ye eklenmesi için Geribildirim değeri.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. Geribildirim (MQUZE).

Sözdizimi: almak için: *feedback & = MQDistributionListItem*.**Feedback**

Ayarlamak için: *MQDistributionListÖğesi*.**Geribildirim** = *geribildirim &*

GroupId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için GroupId . Başlangıç değeri tüm boş değerlerde olur.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 karakter dizgisi

Sözdizimi: almak için: *groupid $= MQDistributionListItem*.**GroupId**

Ayarlamak için: *MQDistributionListItem*.**GroupId** = *groupid $*

GroupIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye eklenmesi için GroupId .

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çiftini temsil eder ve tek karakteri "B" vb. gösterir.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 .. 0".

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 ASCII karakteri yeniden sunan, 48 onaltılı karakterden oluşan dizgi.

Sözdizimi: almak için: *groupidh $= MQDistributionListItem*.**GroupIdHex**

Ayarlamak için: *MQDistributionListItem*.**GroupIdHex** = *groupidh $*

MessageId özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye dahil edilmesi için MessageId . Başlangıç değeri tüm boş değerlerde olur.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 karakter dizgisi

Sözdizimi: almak için: *messageid $= MQDistributionListItem*.**MessageId**

Ayarlamak için: *MQDistributionListItem*.**MessageId** = *messageid $*

MessageIdHex özelliği

Okuma-yazma. Bir kuyruğa konduğunda iletinin MQPMR ' ye dahil edilmesi için MessageId .

Dizilimin her iki karakteri, tek bir ASCII karakterinin onaltılı değerini gösterir. Örneğin, "6" ve "1" karakter çifti, tek karakteri "A", "6" ve "2" karakter çiftini temsil eder ve tek karakteri "B" vb. gösterir.

48 geçerli onaltılı karakter sağlamalısınız.

Başlangıç değeri: "0 .. 0".

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
24 ASCII karakteri simgeleyen 48 onaltılı karakterden oluşan dizgi.

Sözdizimi: almak için: *messageidh $= MQDistributionListItem*.**MessageIdHex**

Ayarlamak için: *MQDistributionListItem*.**MessageIdHex** = *messageidh $*

NextDistributionListItem özelliği

Salt okunur. Bir sonraki dağıtım listesi öğesi nesnesi, aynı dağıtım listesiyle ilişkilendirildi.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
MQDistributionListÖğesi

Sözdizimi: almak için: *set distributionlistitem = MQDistributionListöğesi*.**NextDistributionListItem**

PreviousDistributionListItem özelliği

Salt okunur. Aynı dağıtım listesiyle ilişkilendirilmiş önceki dağıtım listesi öğesi nesnesi.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
MQDistributionListÖğesi

Sözdizimi: almak için: *set distributionlistitem = MQDistributionListItem*.**PreviousDistributionListItem**

QueueManagerAd özelliği

Okuma-yazma. WebSphere MQ kuyruk yöneticisi adı.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
48 karakter dizgisi.

Sözdizimi: almak için: *qmname $= MQDistributionListItem*.**QueueManagerName**

Ayarlamak için: *MQDistributionListItem*.**QueueManagerName** = *qmname $*

QueueName özelliği

Okuma-yazma. WebSphere MQ kuyruk adı.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
48 karakter dizgisi.

Sözdizimi: almak için: *qname $= MQDistributionListItem*.**QueueName**

Ayarlamak için: *MQDistributionListItem*.**QueueName** = *qname $*

ReasonCode özelliği

Salt okunur. Son açma ya da koyma değerine göre belirlenen tamamlanma kodu, sahip olan dağıtım listesi nesnesine ayarlanır.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
Uzun

Değerler:
Bkz. API neden kodları.

- MQCC_OK
- MQCC_UYARı
- MQCC_FAILED

Sözdizimi: almak için: *reasoncode & = MQDistributionListItem*.**ReasonCode**

ReasonName özelliği

Salt okunur. ReasonCodeiçin simgesel ad. Örneğin, "MQRC_QMGR_NOT_AVALABILIR".

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Veri Tipi:
Dizgi

Değerler:
Bkz. API neden kodları.

Sözdizimi: almak için: *reasonname $= MQDistributionListItem*.**ReasonName**

ClearErrorCodes yöntemi

CompletionCode öğesini, hem MQDistributionListöğe sınıfı, hem de MQSession sınıfı için MQCC_NONE ve MQRC_NONE MQRC_NONE olarak sıfırlar.

Tanımlandığı yer:
MQDistributionListÖğe sınıfı

Sözdizimi: Call *MQDistributionListItem*.**ClearErrorCodes**

Sorun giderme

Sağlanan izleme tesisine ilişkin bilgiler, ortak atışlara ilişkin bilgiler ve bunların nasıl saklanmalarına yardımcı olur.

Aşağıdaki bölümde, sağlanan izleme olanağı açıklanır ve bunların önüne geçilmesine ilişkin yardım içeren ortak atışlara ilişkin ayrıntılar açıklanmaktadır:

İzlemenin kullanılması

MQAX, bir sorun olduğunda hizmet kuruluşunun ne olduğunu belirlemesine yardımcı olacak bir izleme olanağını içerir. Bu, MQAX komut dosyanızı çalıştırdığınızda alınan yolları gösterir. Bir sorun yoksa, sistem kaynaklarının gereksiz kullanımını önlemek için izleme kümesiyle çalıştırın.

İzlemeyi denetlemek için ayarladığınız üç ortam değişkeni vardır:

- OMQ_TRACE
- OMQ_TRACE_PATH
- OQ_IZLEME_DÜZEYI

OMQ_TRACE için *any* değerinin belirlendiğini belirten not, izleme yüzgeçicisini açar. OMQ_TRACE ayarı OFF değerine ayarlansa bile, izleme etkin olmaya devam eder.

İzlemeyi kapatmak için, OMQ_TRACE için bir değer belirtmeyin.

1. **Başlat**düğmesini tıklatın.
2. **Control Panel**(Denetim Masası)
3. **System**(Sistem) seçeneğini çift tıklatın
4. **Gelişmiş**düğmesini tıklatın.
5. **Ortam**düğmesini tıklatın.
6. "Kullanıcı değişkenleri (kullanıcı adı)" başlıklı bölümde **Yeni**' yi tıklatın.
7. Değişken adını ve uygun alanlara geçerli bir değer girin ve **Tamam**düğmesini tıklatın.
8. Ortam Değişkenleri penceresini kapatmak için **Tamam** düğmesini tıklatın.
9. Sistem Özellikleri penceresini kapatmak için **Tamam** düğmesini tıklatın.
10. Control Panel (Denetim Masası) penceresini kapatın

İzleme dosyalarının yazılmasını istediğiniz yere karar verirken, yalnızca okumayı değil, diski yazmak için yeterli yetkiye sahip olduğundan emin olun.

İzlemenin açık olması, MQAX 'in çalışmasını yavaşlatır, ancak ActiveX ya da WebSphere MQ ortamlarınızın performansını etkilemez. Artık bir izleme dosyasına gerek kalmadığında, dosyayı silebilirsiniz.

OMQ_TRACE değişkeninin durumunu değiştirmek için çalışan MQAX 'i durdurmanız gerekir.

İzleme dosyası adı ve dizini

İzleme kütüğü adı OMQnnnnn.trcbiçimini alır; burada nnnnn, o sırada çalışmakta olan ActiveX işleminin tanıtıcısıdır.

Çizelge 157. Komutlar ve etkileri	
Komut	**Efekt**
SET OMQ_TRACE_PATH = sürücü: \directory	İzleme dosyasının yazılacağı izleme dizinini belirler.
SET OMQ_TRACE_PATH =	OMQ_PATH ortam değişkenini yürürlükteki çalışma dizini (ActiveX başlatıldığında) kaldırıldığında kaldırır.
ECHO %OMQ_TRACE_PATH%	Windows' ta izleme dizininin yürürlükteki ayarını görüntüler.
SET OMQ_TRACE = xxxxxxxx	Bu, izleme olarak izlemeyi ayarlar. İzlemeyi '=' işaretinden sonra bir ya da daha çok karakter koyarak açmanızı sağlar. Örnek: SET OMQ_TRACE=yes SET OMQ_TRACE = no. SET. Bu örneklerin her ikisinde de izleme ON (Açık) olarak ayarlanacaktır. Bu yalnızca tek bir window/session için etkilidir.
SET OMQ_TRACE=	İzlemeyi KAPATIN
ECHO %OMQ_TRACE%	Windowsortam değişkenine ilişkin içeriği görüntüler.
Belirle	Windows' ta bulunan tüm ortam değişkenlerinin içeriğini görüntüler.
SET OMQ_TRACE_LEVEL = 9	İzleme düzeyini 9 olarak ayarlar. 9 'dan büyük değerler, izleme dosyasında ek bilgi üretmez.

WebSphere MQ Automation Classes for ActiveX komut dosyası başarısız olduğunda

If your WebSphere MQ Automation Classes for ActiveX script fails, there are a number of sources of information.

İlk hata belirtisi raporu

İzleme olanından bağımsız olarak, beklenmeyen ve iç hatalar için ilk hata belirtisi raporu üretilebilir.

Bu rapor, OMQnnnnn.fdcadlı bir dosyada bulunur; burada nnnnn, o sırada çalışmakta olan ActiveX işleminin numarasıdır. Bu dosyayı, ActiveX ' u başlattığınız çalışma dizininde ya da OMQ_PATH ortam değişkeninde belirtilen yolda bulursınız.

Diğer bilgi kaynakları

WebSphere MQ , ilgili altyapıya bağlı olarak çeşitli hata günlükleri ve izleme bilgileri sağlar. Windows NT uygulama olay günlüğünüze bakın.

Neden kodları

WebSphere MQ MQI için belgelenenlere ek olarak aşağıdaki neden kodları da olabilir. Diğer kodlar için WebSphere MQ uygulama olay günlüğünüze bakın.

Çizelge 158. Neden kodları ve ne anlama geldikleri	
Neden Kodu	**Açıklama**
MQRC_LIBRARY_LOAD_ERROR (6000)	WebSphere MQ kitaplıklarından biri ya da daha fazlası yüklenemedi. Tüm WebSphere MQ kitaplıklarının, kullanmakta olduğunuz sistemde doğru arama yolunda olduğunu doğrulayın. Örneğin, WebSphere MQ kitaplıklarını içeren dizinlerin PATH değişkeninde olduğundan emin olun.
MQRC_CLASS_LIBRARY_ERROR (6001)	WebSphere MQ classlibrary çağrılarından biri beklenmeyen bir ReasonCode/CompletionCodeçağrısını döndürdü. Ayrıntılar için First Failure Symptom Raporuna bakın. Kullanılmakta olan son yöntemi/ özelliği ve sınıfı not alın ve sorunun IBM Destek birimine bildirmesini bildirin.

Çizelge 158. Neden kodları ve ne anlama geldikleri (devamı var)

Neden Kodu	Açıklama
MQRC_STRING_LENGTH_TOO_BIG (6002)	İleti arabelleğindeki uzunluğu 65,535 byte 'tan büyük bir UTF biçim dizgisi yazmak için girişimde bulunuldu.
MQRC_WRITE_VALUE_ERROR (6003)	Aralık dışında bir değer kullanılır; örneğin, msg.WriteByte (240).
MQRC_PACKED_DECIMAL_ERROR (6004)	İleti arabelleğinden paketlenmiş onlu sayıyı okuma girişiminde bulunuldu, ancak veri işaretçisinde veri geçerli bir paketlenmiş veri biçiminde değil.
MQRC_FLOAT_CONVERSION_ERROR (6005)	İleti arabelleğinden tek ya da çift kayan noktalı sayı okuma girişiminde bulunuldu, ancak veri işaretçisinde veri imleci uygun bir kayar noktalı biçimde değil.
MQRC_REOPEN_EXCL_INPUT_ERROR (6100)	Açık bir nesne, doğru **OpenOptions** ' a sahip değildir ve bir ya da daha fazla ek seçenek gerektirir. Örtük bir yeniden açma gerekli, ancak kapatma engellendi. Set the **OpenOptions**explicitly to cover all eventualities so that implicit reopening is not required. Kuyruk dışlayıcı giriş ve kapanış için açık olduğu için, kapatma işlemi engellendi; bu kuyruk, diğer kişilerin kuyruğa erişmesi için bir fırsat penceresi sunacaktır.
MQRC_REOPEN_INQUIRE_ERROR (6101)	Açık bir nesne doğru OpenOptions ' a sahip değildir ve bir ya da daha fazla ek seçenek gerektirir. Örtük bir yeniden açma gerekli, ancak kapatma engellendi. OpenOptions ' i MQOO_SORGULAMAK için açık bir şekilde ayarlayın. Kapatılmadan önce nesnenin bir ya da daha fazla özelliği için dinamik olarak denetlenmesi gerektiğinden ve OpenOptions MQOO_SORGULAMASINI içermediğinden, kapatma işlemi engellendi.
MQRC_REOPEN_SAVED_CONTEXT_ERR (6102)	Açık bir nesne doğru OpenOptions ' a sahip değildir ve bir ya da daha fazla ek seçenek gerektirir. Örtük bir yeniden açma gerekli, ancak kapatma engellendi. Set the OpenOptions explicitly to cover all eventualities so that implicit reopening is not required. Kuyruk MQOO_SAVE_ALL_CONTEXT ile açık olduğu ve daha önce yıkıcı bir Alma işlemi gerçekleştirildiği için kapatma engellendi. Bu, alıkonan durum bilgilerinin açık kuyrukla ilişkilendirilmesine neden oldu ve bu bilgiler kapatılarak yok edilecek.
MQRC_REOPEN_TEMPORARY_Q_ERROR (6103)	Açık bir nesne, doğru **OpenOptions** ' a sahip değildir ve bir ya da daha fazla ek seçenek gerektirir. Örtük bir yeniden açma gerekli, ancak kapatma engellendi. Set the **OpenOptions**explicitly to cover all eventualities so that implicit reopening is not required. Kuyruk, kapatılarak yok edilecek MQQDT_TEMPORARY_DYNAMIC tanımlama tipinde yerel bir kuyruk olduğu için kapatma engellendi.
MQRC_ATTRIBUTE_LOCKED (6104)	Nesne açıkken, bir nesnenin değerini ya da özniteliğini değiştirmek için girişimde bulunuldu. **AlternateUserTnt**gibi bazı öznitelikler, bir nesne açıkken değiştirilemez.
MQRC_CURSOR_NOT_VALID (6105)	Açık bir kuyruk için göz atma imleci, örtük bir yeniden açma tarafından son kullanılandan sonra geçersiz kılındı. Set the OpenOptions explicitly to cover all eventualities so that implicit reopening is not required.
MQRCENCODING_ERROR (6106)	Sonraki ileti öğesinin kodlaması, okumak için MQENC_NATIVE olmalıdır.
MQRC_STRUCID_ERROR (6107)	Veri işaretçisinden başlayarak, 4 karakterden türetilen sonraki ileti öğesine ilişkin tanıtıcı eksik ya da öğenin okunmakta olduğu değişken tipiyle tutarsız.
MQRC_NULL_POINTER (6108)	Boş değerli (null) olmayan bir işaretçinin gerekli ya da örtük olarak bulunduğu boş değerli gösterge belirtildi. Bu durum, VBA ' dan çağrılara parametre olarak kullanılan WebSphere MQ nesnelerine ilişkin belirtik bildirimler kullanılmasından kaynaklanabilir (örneğin, Nesne olarak dim msg, ok, MqMessage gibi dim, sorunlara neden olabilir). Örneğin, Excel 'de, q tanımlı ve set dim msg olarak MqMessageq.put msg, reasonCode MQRC_NULL_POINTER değeri verir. VisualBasic' ten doğru şekilde çalışır.
MQRC_NO_CONNECTION_REFERCE (6109)	**MQQueue** nesnesi, **MQQueueManager**bağlantısıyla bağlantısını kaybetti. **MQQueueManager**bağlantısı kesilirse bu durum ortaya çıkar. **MQQueue** nesnesini silin.

Çizelge 158. Neden kodları ve ne anlama geldikleri (devamı var)

Neden Kodu	Açıklama
MQRC_NO_BUFFER (6110)	Kullanılabilir arabellek yok. Bir **MQMessage** nesnesi için, nesne durumunda oluşmaması gereken iç tutarsızlığı belirten bir nesne ayrılamaz.
MQRC_BINARY_DATA_LENGTH_ERROR (6111)	İkili verilerin uzunluğu, hedef özniteliğin uzunluğuna göre tutarsız. Sıfır, tüm öznitelikler için doğru bir uzunlukdır. 24 is a correct length for a **CorrelationId** and for a **MessageId** 32 is a correct length for an **AccountingToken**
MQRC_BUFFER_NOT_AUTOMAKS (6112)	Kullanıcı tanımlı ve yönetilen arabellekler yeniden boyutlandırılamaz. İleti arabellekleri sistem tarafından yönetildiğinden, bu bir iç tutarsızlığı gösterir.
MQRC_INSUFFICIENT_BUFFER (6113)	Veri işaretçisinin isteği yerine getirmesinin ardından kullanılabilir arabellek alanı yetersiz. Bunun nedeni, arabelleğin yeniden boyutlandırılamaması olabilir.
MQRC_INSUFFICIENT_DATA (6114)	Veri işaretçisi okuma isteğini yerine getirmek için yeterli veri yok. Arabelleği doğru boyutlara indirin ve verileri yeniden okuyun.
MQRC_DATA_TRUNCATED (6115)	Veriler bir arabellekten diğerine kopyalanırken kesildi. Bunun nedeni, hedef arabelleğin yeniden boyutlandırılamaması ya da bir ya da başka bir arabelleğin adreslenmesiyle ilgili bir sorun olması ya da bir arabellek daha küçük bir değiştirmeyle küçülmekte olduğu için olabilir.
MQRC_ZERO_LENGTH (6116)	Pozitif bir uzunluğun gerekli ya da örtük olduğu bir yere sıfır (sıfır) uzunluk değeri verildi.
MQRC_NEGATIVE_LENGTH (6117)	Sıfır ya da pozitif bir uzunluğun gerekli olduğu yerde negatif bir uzunluk belirtildi.
MQRC_NEGATIVE_OFFSET (6118)	Sıfır ya da pozitif göreli konum gerekli olduğunda negatif göreli konum sağlandı.
MQRC_INCONSISTENT_BIçIMI (6119)	Sonraki ileti öğesinin biçimi, öğenin okunmakta olduğu değişkenle tutarsız.
MQRC_INCONSSTENT_OBJECT_STATE (6120)	Bu nesne ile açık olan ve bağlı olmayan MQQueueManager nesnesi arasında bir tutarsızlık var.
MQRC_CONTEXT_OBJECT_NOT_VALID (6121)	MQPutMessageSeçenekleri bağlam başvurusu, geçerli bir MQQueue nesnesine gönderme yapmamaktadır. Obje previoulsy yok edildi.
MQRC_CONTEXT_OPEN_ERROR (6122)	MQPutMessageSeçenekleri bağlam başvurusu, bağlam oluşturmak için açılamayabilecek bir MQQueue nesnesine gönderme yapıyor. Bunun nedeni, MQQueue nesnesinin uygun olmayan açık seçenekleri olması olabilir. Hatanın nedenini belirlemek için başvurulan nesne neden kodunu inceleyin.
MQRC_STRUC_LENGTH_ERROR (6123)	İç veri yapısının uzunluğu, içeriğiyle tutarlı değil. Bir MQRMH için, uzunluk değişmez alanları ve tüm göreli konum verilerini içermemek için yeterli olur.
MQRC_NOT_CONNECTED (6124)	Bir kuyruk yöneticisine gerekli bir bağlantı kullanılamadığı için bir yöntem başarısız oldu ve örtük olarak bir bağlantı kurulamaz.
MQRC_NOT_OPEN (6125)	Bir WebSphere MQ nesnesi açık olmadığı için bir yöntem başarısız oldu ve açma örtük olarak gerçekleştirilemez.
MQRC_DISTRIBUTION_LIST_BOŞ (6126)	Dağıtım listesinde MQDistributionList öğe nesnesi olmadığı için bir MQDistributionList açılamadı. Düzeltici işlem: Dağıtım listesine en az bir MQDistributionListöğe nesnesi ekleyin.
MQRC_INCONSISTENT_OPEN_SEçENEKLERI (6127)	Nesne açık olduğundan ve açma seçenekleri gerekli işlemle tutarsız olduğundan bir yöntem başarısız oldu. Düzeltici işlem: Uygun açık seçeneklerle nesneyi açın ve yeniden deneyin.
MQRC_WRONG_VERSION (6128)	Belirtilen ya da saptanan bir sürüm numarası yanlış ya da desteklenmediği için yöntem başarısız oldu.

Kod düzeyi aracı

Hangi kod düzeyini kurduğunuz IBM Hizmet Ekibi tarafından size sorabilirsiniz.

Bunu öğrenmek için 'MQAXLEV' yardımcı programını çalıştırın.

Komut isteminden, MQAX200.dll ' i içeren dizine geçin ya da tam yol uzunluğunu ekleyin ve şunu girin:

```
MQAXLev MQAX200.dll > MQAXLEV.OUT
```

Burada MQAXLEV.OUT , çıkış dosyasının adıdır.

Bir çıkış dosyası belirtmezseniz, ayrıntı ekranda görüntülenir.

Aşağıdaki örnekte, kod düzeyi aracından örnek bir çıkış dosyası ayrıntılı olarak açıklanmıştır:

Kod düzeyi aracından örnek çıkış dosyası

```
5639-B43 (C) Copyright IBM Corp. 1996, 2024. ALL RIGHTS RESERVED.
     ***** Code Level is 5.1 ***** lib/mqole/mqole.cpp, mqole, p000, p000 L981119 1.8 98/08/21
lib/mqlsx/gmqdyn0a.c, mqlsx, p000, p000 L990212  1.6 99/02/11 16:40:24
lib/mqlsx/pc/gmqdyn1p.c, mqlsx, p000, p000 L990212  1.6 99/02/11 16:44:14
lib/mqlsx/xmqcsa.c, mqole, p000, p000 L990216             1.3 99/02/15 13:24:34
lib/mqlsx/xmqfdca.c, mqlsx, p000, p000 L990212               1.3 99/02/11 16:40:35
lib/mqlsx/xmqtrca.c, mqlsx, p000, p000 L990212            1.5 99/02/11 16:12:02
lib/mqlsx/xmqutila.c, mqlsx, p000, p000 L990212           1.3 99/02/11 16:40:40
lib/mqlsx/xmqutl1a.c, mqlsx, p000, p000 L990212           1.4 99/02/11 16:40:30
lib/mqlsx/xmqcnv1a.c, mqlsx, p000, p000 L990212           1.9 99/02/11 16:40:56
lib/mqlsx/xmqmsg.c, mqole, p000, p000 L990219             1.11 99/02/18 12:12:59
```

MQAI ' yeActiveX arabirimi

COM arabirimlerine ve MQAI ' de kullanımlarına ilişkin kısa bir genel bakış için bkz. “Component Object Model Interface olanağının kullanılması (ActiveXiçin WebSphere MQ Automation Sınıfları)” sayfa 992.

MQAI, uygulamaların PCF için gereken değişken uzunluklu arabellekleri doğrudan edinmeden ve biçimlendirmeden Programlanır Komut Biçimi (PCF) komutları oluşturmasına ve göndermesine olanak sağlar. MQAI ile ilgili daha fazla bilgi için bkz. WebSphere MQ Yönetim Arabirimi 'ne (MQAI) giriş. MQAI ActiveX MQBag sınıfı, MQAI tarafından desteklenen veri torbalarını, COM nesnelerinin yaratılmasını destekleyen herhangi bir dilde (örneğin, Visual Basic, C + +, Java ve diğer ActiveX komut dosyası istemcileri) kullanabilecek bir şekilde kapsüller.

MQAI ActiveX arabirimi, MQI ' ye bir COM arabirimi sağlayan MQAX sınıflarıyla birlikte kullanılır. MQAX sınıflarına ilişkin ek bilgi için bkz. “ActiveX dışı uygulamalara erişen MQAX uygulamalarının tasarlanması” sayfa 993.

ActiveX arabirimi, MQBag adında tek bir sınıf sağlar. Bu sınıf, MQAI veri torbaları yaratmak için kullanılır; özellikleri ve yöntemleri, her bir çanta içinde veri öğeleri yaratmak ve bu öğeleri kullanarak çalışmak için kullanılır. MQBag Execute yöntemi, çanta verilerini bir WebSphere MQ kuyruk yöneticisine bir PCF iletisi olarak gönderir ve yanıtları toplar.

MQBag sınıfı, özellikleri ve yöntemleri hakkında daha fazla bilgi için bkz. “MQBag sınıfı” sayfa 1067.

PCF iletisi, belirlenen kuyruk yöneticisi nesnesine, isteğe bağlı olarak belirlenen istek ve yanıt kuyruklarını kullanarak gönderilir. Yanıtlar yeni bir MQBag nesnesi içinde döndürülür. Komutların ve yanıtların tam kümesi, Programlanır Komut Biçimlerinin Tanımlarıbaşlıklı konu altında açıklanmıştır. Komutlar, uygun istek ve yanıt kuyruklarını seçerek, WebSphere MQ ağındaki herhangi bir kuyruk yöneticisine gönderilebilir.

MQBag sınıfı

MQBag sınıfı, gerekli olduğu şekilde MQBag nesneleri yaratmak için kullanılır. Somutlaştırıldığında, MQBag sınıfı yeni bir MQBag nesne başvurusu döndürür.

Visual Basic 'te bir MQBag nesnesi yaratmak için aşağıdaki komutu girin:

```
Dim mqbag As MQBag
Set mqbag = New MQBag
```

MQBag Özelliği

MQBag nesnelerine ilişkin özellikler aşağıdaki listede açıklanmıştır:

- “Öğe özelliği” sayfa 1068.
- “Sayı özelliği” sayfa 1069.
- “Seçenekler özelliği” sayfa 1070.

MQBag yöntemleri

MQBag nesnelerine ilişkin yöntemler aşağıdaki listede açıklanmıştır:

- “Yöntem ekle” sayfa 1071.
- “AddInquiry yöntemi” sayfa 1071.
- “Yöntemi temizle” sayfa 1071.
- “Yöntemi yürüt” sayfa 1072.
- “FromMessage yöntemi” sayfa 1072.
- “ItemType yöntemi” sayfa 1073.
- “Yöntemi kaldır” sayfa 1074.
- “Seçici yöntemi” sayfa 1074.
- “ToMessage yöntemi” sayfa 1075.
- “Yöntem kes” sayfa 1075.

Hata İşleme

MQBag nesnesindeki bir işlem sırasında, temeldeki MQAX ya da MQAI nesnesi tarafından torbaya döndürülen hatalar da içinde olmak üzere bir hata saptanırsa, hata kural dışı durumu ortaya çıktı. MQBag sınıfı, COM ISupportErrorbilgi arabirimini destekler; bu nedenle, hata işleme yordamınıza aşağıdaki bilgiler kullanılabilir:

- Hata numarası: saptanan hata ve bir COM olanağı kodu için WebSphere MQ neden kodundan oluşur. Tesis alanı, COM için standart olarak, hatanın sorumluluk alanını gösterir. WebSphere MQ tarafından saptanan hatalar için her zaman ABITYY_ITF olur.
- Hata kaynağı: hatayı saptayan nesnenin tipini ve sürümünü belirtir. MQBag işlemleri sırasında saptanan hatalar için, hata kaynağı her zaman MQBag.MQBag1' dir.
- Hata açıklaması: WebSphere MQ neden kodu için simgesel adı veren dizgi.

Hata bilgilerine erişim, komut dosyası yazma dilinize bağlıdır; örneğin, Visual Basic 'te bilgiler Err nesnesinde döndürülür ve WebSphere MQ neden kodu, Err.Number' tan sabit vbObjectHatasının çıkarılmasıyla elde edilir.

ReasonCode = Err.Number - vbObjectHatası

MQBag Execute iletisi bir PCF iletisi gönderirse ve bir yanıt alınırsa, gönderilen komut başarısız olsa da işlem başarılı olarak kabul edilir. Bu durumda, yanıt torbasının kendisi, Programlanabilir Komut Biçimlerinin Tanımlamaları içinde açıklandığı şekilde tamamlanma ve hata neden kodlarını içerir.

Öğe özelliği

Amaç

Öğe özelliği, bir çantadaki bir öğeyi temsil eder. Bir öğenin değerini ayarlamak ya da sorgulamak için kullanılır. Bu özelliğin kullanılması, aşağıdaki MQAI çağrılarına karşılık gelir:

- "mqSetDizgisi"
- "mqSetTamsayı"
- "mqInquireTamsayı"
- "mqInquireDizgi"
- "mqInquireBag"

in the Programlanabilir komut biçimleri başvurusu.

Biçim

Öğe (Seçici, ItemIndex, Değer)

Parametreler

Selector **(VARIANT)-giriş**
Ayarlanacak ya da sorgulanacak öğenin seçicisi.

Bir öğeyle ilgili araştırma sırasında, MQSEL_ANY_USER_SELECTOR varsayılan değerdir. Bir öğe ayarlarken, varsayılan olarak MQIA_LIST ya da MQCA_LIST değeri olur.

`Selector` uzun tipte bir tipse, MQRC_SELECTOR_TYPE_ERROR sonuçları.

Bu parametre isteğe bağlıdır.

ItemIndex **(LONG)-giriş**
Bu değer, belirlenecek ya da sorgulanacak belirlenen seçiciye ilişkin öğenin oluşumunu tanımlar. MQIND_NONE varsayılan değerdir.

Bu parametre isteğe bağlıdır.

Value **(VARIANT)-giriş/çıkış**
Döndürülelacak değer ya da ayarlanacak değer. Bir öğe sorulması sırasında, dönüş değeri long, string ya da MQBag tipinde olabilir. Ancak, bir öğe ayarlanırken, değer long ya da string tipinde olmalıdır; değilse, MQRC_ITEM_VALUE_ERROR sonuçları.

Visual Basic Dili Çağırma

Bir çantanın içindeki bir öğenin değerini ne zaman sorup soruladığınızda:

```
Value = mqbag[.Item]([Selector],
[ItemIndex])
```

MQBag başvuruları için:

```
Set abag = mqbag[.Item]([Selector].
[ItemIndex])
```

Bir öğenin bir çantadaki değerini ayarlamak için:

```
mqbag[.Item]([Selector],
[ItemIndex]) = Value
```

Sayı özelliği

Amaç

Count özelliği, bir çanta içindeki veri öğelerinin sayısını temsil eder. Bu özellik, Programlanabilir komut biçimleri başvurusuiçindeki "mqCountÖğeleri" MQAI çağrısına karşılık gelir.

Biçim

Sayı (*Selector, Value*)

Parametreler

***Selector* (VARIANT)-giriş**
Sayıma dahil edilecek veri öğelerinin seçicisi.

MQSEL_ALL_USER_SELECTORS, varsayılan değer olarak kullanılan varsayılan değerdir.

`Selector` uzun tipli değilse, MQRC_SELECTOR_TYPE_ERROR döndürülür.

***Value* (LONG)-çıkış**
The number of items in the bag included by the *Selector*.

Visual Basic Dili Çağırma

Bir çantadaki öğelerin sayısını döndürmek için:

```
ItemCount = mqbag.Count([Selector])
```

Seçenekler özelliği

Amaç

Seçenekler özelliği, bir çantanın kullanımına ilişkin seçenekleri ayarlar. This property corresponds to the `Options` parameter of the MQAI call, "mqCreateBag," in the Programlanabilir komut biçimleri başvurusu.

Biçim

Seçenekler (*Options*)

Parametreler

***Options* (LONG)-giriş/çıkış**
Çanta seçenekleri.

Not: The bag options must be set ***önce*** data items are added to or set within the bag. Çanta boş değilse seçenekler değiştirilirse, MQRC_OPTIONS_ERROR sonuçları gelir. Bu, çanta daha sonra temizlenmiş olsa bile geçerlidir.

Visual Basic Dili Çağırma

Bir çantanın içindeki bir öğenin seçenekleri ne zaman sorulabiliyor:

```
Options = mqbag.Options
```

Çantadaki bir öğenin bir seçeneğini ayarlamak için:

```
mqbag.Options = Options
```

MQBag yöntemleri

MQBag nesnelerine ilişkin yöntemler, aşağıdaki sayfalar üzerinden açıklanır.

Yöntem ekle

Amaç

Add yöntemi, bir çantaya veri öğesi ekler. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki MQAI çağrılarına, "mqAddInteger" ve "mqAddDizgisi" öğesine karşılık gelir.

Biçim

Ekle (*`Value, Selector`*)

Parametreler

***`Value`* (VARIANT)-giriş**
Veri öğesinin tamsayı ya da dizgi değeri.

***`Selector`* (VARIANT)-giriş**
Eklenecek öğeyi tanımlayan seçici.

`Value`etipine bağlı olarak, varsayılan olarak MQIA_LIST ya da MQCA_LIST olur. `Selector` parametresi uzun tipte değilse, MQRC_SELECTOR_TYPE_ERROR sonuçları.

Visual Basic Dili Çağırma

Bir torbaya öğe eklemek için:

```
mqbag.Add(Value,[Selector])
```

AddInquiry yöntemi

Amaç

AddInquiry yöntemi, bir SORGULAMA komutu yürütmek için bir denetim paketi gönderildiğinde döndürülelacak özniteliği belirten bir seçici eklemenizi sağlar. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqAddSorgusu" MQAI çağrısına karşılık gelir.

Biçim

AddInquiry *`(Inquiry`*)

Parametreler

***`Inquiry`* (LONG)-giriş**
SORGULAMA denetim komutu tarafından döndürülebilmek için WebSphere MQ özniteliğinin seçicisi.

Visual Basic Dili Çağırma

AddInquiry yöntemini kullanmak için:

```
mqbag.AddInquiry(Inquiry)
```

Yöntemi temizle

Amaç

Temizleme yöntemi, bir torbadan tüm veri öğelerini siler. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqClearBag" MQAI çağrısına karşılık gelir.

Biçim

Temizle

Visual Basic Dili Çağırma

Bir paketten tüm veri itmelerini silmek için:

```
mqbag.Clear
```

Yöntemi yürüt

Amaç

Execute y "ntemi, komut sunucusuna bir denetim komutu iletisi g" nderip, herhangi bir yanıt iletisi için bekler. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçinde MQAI çağrısına ("mqExecute,") karşılık gelir.

Biçim

Execute *(QueueManager, Command, OptionsBag, RequestQ, ReplyQ, ReplyBag*)

Parametreler

***QueueManager* (MQQueueManager)-giriş**
Uygulamanın bağlandığın kuyruk yöneticisi.

***Command* (LONG)-giriş**
Yürütülecek komut.

***OptionsBag* (MQBag)-giriş**
Aramanın işlenmesini etkileyen seçenekleri içeren çanta.

***RequestQ* (MQQueue)-giriş**
Denetim komut iletisinin konacağı kuyruk.

***ReplyQ* (MQQueue)-giriş**
Herhangi bir yanıt iletisinin alındığı kuyruk.

***ReplyBag* (MQBag)-çıkış**
Yanıt iletilerinden alınan verileri içeren bir çanta başvurusu.

Visual Basic Dili Çağırma

Bir denetim komutu iletisi göndermek ve yanıt iletilerini beklemek için:

```
Set ReplyBag = mqbag.Execute(QueueManager, Command,
[OptionsBag],[RequestQ],[ReplyQ])
```

FromMessage yöntemi

Amaç

FromMessage yöntemi, verileri bir iletiden çantaya yükler. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqBufferToBag" MQAI çağrısına karşılık gelir.

Biçim

FromMessage *(Message, OptionsBag)*

Parametreler

***Message* (MQMessage)-giriş**
Dönüştürülecek verileri içeren ileti.

***OptionsBag* (MQBag)-giriş**
Aramanın işlenmesini denetleyen seçenekler.

Visual Basic Dili Çağırma

Bir iletiden bir çantaya veri yüklemek için:

```
mqbag.FromMessage(Message,[OptionsBag])
```

ItemType yöntemi

Amaç

ItemType yöntemi, bir çantada belirtilen bir öğede değerin tipini döndürür. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqInquireItemInfo," MQAI çağrısına karşılık gelir.

Biçim

ItemType *(Selector, ItemIndex, ItemType)*

Parametreler

***Selector* (VARIANT)-giriş**
Sorgulanacak öğeyi tanımlayan seçici.

MQSEL_ANY_USER_SELECTOR varsayılan değerdir. `Selector` parametresi uzun tipte değilse, MQRC_SELECTOR_TYPE_ERROR sonuçları.

***ItemIndex* (LONG)-giriş**
Sorgulanacak öğelerin dizini.

MQIND_NONE varsayılan değerdir.

***ItemType* (LONG)-çıkış**
Belirtilen öğenin veri tipi.

Not: `Selector` parametresi, `ItemIndex` parametresi ya da her ikisi de belirtilmelidir. Her iki değiştirge de varsa, MQRC_PARAMETER_MISSING sonuçları.

Visual Basic Dili Çağırma

Bir değerin tipini döndürmek için:

```
ItemType = mqbag.ItemType([Selector],
[ItemIndex])
```

Yöntemi kaldır

Amaç

Kaldırma (Remove) yöntemi, bir öğeyi çantadan siler. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqDeleteItem" adlı MQAI çağrısına karşılık gelir.

Biçim

Remove *(Selector, ItemIndex)*

Parametreler

***Selector* (VARIANT)-giriş**
Silinecek öğeyi tanımlayan seçici.

MQSEL_ANY_USER_SELECTOR varsayılan değerdir. `Selector` parametresi uzun tipte değilse, MQRC_SELECTOR_TYPE_ERROR sonuçları.

***ItemIndex* (LONG)-giriş**
Silinecek öğenin dizini.

MQIND_NONE varsayılan değerdir.

Not: `Selector` parametresi, `ItemIndex` parametresi ya da her ikisi de belirtilmelidir. Her iki değiştirge de varsa, MQRC_PARAMETER_MISSING sonuçları.

Visual Basic Dili Çağırma

Çantadan bir öğeyi silmek için:

```
mqbag.Remove([Selector],[ItemIndex])
```

Seçici yöntemi

Amaç

Seçici yöntemi, bir çanta içinde belirtilen bir öğenin seçicisini döndürür. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqInquireItemInfo" MQAI çağrısına karşılık gelir.

Biçim

Seçici *(Selector, ItemIndex, OutSelector)*

Parametreler

***Selector* (VARIANT)-giriş**
Sorgulanacak öğeyi tanımlayan seçici.

MQSEL_ANY_USER_SELECTOR varsayılan değerdir. `Selector` parametresi uzun tipte değilse, MQRC_SELECTOR_TYPE_ERROR sonuçları.

***ItemIndex* (LONG)-giriş**
Sorgulanacak öğenin dizini.

MQIND_NONE varsayılan değerdir.

***OutSelector* (VARIANT)-çıkış**
Belirtilen öğenin seçicisi.

Not: `Selector` parametresi, `ItemIndex` parametresi ya da her ikisi de belirtilmelidir. Her iki değiştirge de varsa, MQRC_PARAMETER_MISSING sonuçları.

Visual Basic Dili Çağırma

Bir öğenin seçicisini döndürmek için:

```
OutSelector = mqbag.Selector([Selector],
[ItemIndex])
```

ToMessage yöntemi

Amaç

ToMessage yöntemi, bir MQMessage nesnesine başvuru döndürür. Başvuru, bir çantadan veri içeriyor. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqBagToBuffer," MQAI çağrısına karşılık gelir.

Biçim

ToMessage *(OptionsBag, Message)*

Parametreler

***OptionsBag* (MQBag)-giriş**
Yöntemin işlenmesini denetleyen seçenekleri içeren bir çanta.

***Message* (MQMessage)-çıkış**
Çantadan veri içeren bir MQMessage nesnesi başvurusu.

Visual Basic Dili Çağırma

ToMessage Yöntemini kullanmak için:

```
Set Message = mqbag.ToMessage([OptionsBag])
```

Yöntem kes

Amaç

Kısaltma yöntemi, bir çantadaki kullanıcı öğelerinin sayısını azaltır. Bu yöntem, Programlanabilir komut biçimleri başvurusuiçindeki "mqTruncateBag" adlı MQAI çağrısına karşılık gelir.

Biçim

Truncate *(ItemCount)*

Parametreler

***ItemCount* (LONG)-giriş**
Kesilme gerçekleştikten sonra, çantada kalacak kullanıcı öğelerinin sayısı.

Visual Basic Dili Çağırma

Bir çantadaki kullanıcı öğelerinin sayısını azaltmak için:

```
mqbag.Truncate(ItemCount)
```

ActiveX Starter örnekleri için WebSphere MQ Automation Sınıfları hakkında

Bu ek, ActiveX Starter örnekleri için WebSphere MQ Otomasyon Sınıflarını açıklar ve bunları nasıl kullanacağını açıklar.

WebSphere MQ for Windows , aşağıdaki Visual Basic örnek programlarını sağlar:

- MQAXTRIV.VBP
- MQAXBSRV.VBP
- MQAXDLST.VBP
- MQAXCLSS.VBP

Bu örnekler Visual Basic 4 ya da Visual Basic 5 üzerinde çalışır. Bunları dizinde bulacaksınız ... \tools\mqax\samples\vb.

Aynı dizinde ayrıca Microsoft Excel ve html için örnekler de bulabilirsiniz. Bu bilgiler şunlardır:

- MQAX.XLS
- MQAXTRIV.XLS
- MQAXTRIV.HTM

Not: If using Visual Basic 5, you **gerekir** select and install Visual Basic component grid32.ocx.

Örneklerde gösterilen nedir?

Örnekler, ActiveX için WebSphere MQ Automation Sınıflarının nasıl kullanılacağını gösterir:

- Kuyruk yöneticisine bağlan
- Kuyruğa erişim
- Kuyruğun üzerine ileti koy
- Kuyruktan ileti al

Visual Basic (Visual Basic) örneğinin merkezi bir bölümü aşağıdaki sayfalarda g " nderilir.

“Örnekleri çalıştırma hazırlığı yapılıyor” sayfa 1077 ve

“Örneklerde işlenirken hata oluştu” sayfa 1077

ActiveX Starter örneklerinin çalıştırılması

Before you run the WebSphere MQ Automation Classes for ActiveX Starter samples check that you have a default queue manager running and that you have created the required queue definitions. Kuyruk yöneticisi yaratıp çalıştırıp kuyruk yaratılmasına ilişkin ayrıntılar için Yönetmedosyasına bakın. The sample uses the queue SYSTEM.DEFAULT.LOCAL.QUEUE which should be defined on any normally set up WebSphere MQ server.

Veri torbalarını kullanmanın farklı yolları aşağıdaki listede gösterildiği gibidir:

- Kuyruk yöneticisine bağlan
- Kuyruğa erişim
- Kuyruğun üzerine ileti koy
- Kuyruktan ileti al

For information on the MQAX starter samples for Microsoft Basic Version 4 or later, see “MQAXTRIV örneğinin çalıştırılması” sayfa 1077

Kuyruk yöneticileri ve kuyruk nesneleri özelliklerine ve yöntemlerine göz atmanıza olanak tanıyan bir örnek hakkında bilgi almak için bkz. “MQAXCLSS örneğinin başlatılması” sayfa 1079

MQAXDLST örneğine ilişkin bilgi edinmek için “MQAXDLST örneği” sayfa 1079

For information on running the MQAX starter sample for Microsoft Excel 95 or later, MQAXTRIV.XLS, see “MQAXTRIV.XLS örneği” sayfa 1079.

Banka gösterisini MQAX.XLSile çalıştırma hakkında bilgi için bkz. “Running the Bank demonstration with MQAX.XLS” sayfa 1079

ActiveX uyumlu bir WWW tarayıcısını kullanarak başlangıç örneğine ilişkin bilgi için bkz. “ActiveX uyumlu bir WWW tarayıcısını kullanarak başlangıç örneği” sayfa 1079

Örnekleri çalıştırma hazırlığı yapılıyor

Örnekleri çalıştırmak için, çalıştırmak istediğiniz örneklerin hangisine bağlı olarak aşağıdakilerden birine gerek duyarsınız.

- Microsoft Visual Basic Sürüm 4 (ya da üstü)
- Microsoft Excel 95 (ya da üstü)
- Bir Web tarayıcısı

Ayrıca aşağıdakiler de gerekir:

- Çalışmakta olan bir WebSphere MQ kuyruk yöneticisi.
- Önceden tanımlanmış bir WebSphere MQ kuyruğu.

Örneklerde işlenirken hata oluştu

Most of the samples provided in the WebSphere MQ Automation Classes for ActiveX package exhibit little or no error handling. Hata işleme hakkında daha fazla bilgi için bkz. “Hata işleme” sayfa 997.

MQAXTRIV örneğinin çalıştırılması

1. Kuyruk yöneticisini başlatın.
2. Windows Explorer ya da File Manager'da örneğin, MQAXTRIV.VBP (Visual Basic Proje dosyası) ve dosyayı açın.

   Visual Basic programı, MQAXTRIV.VBP.
3. Visual Basic 'te, numuneyi çalıştırmak için 5 (F5) tuşuna basın.
4. Pencere biçiminde, "MQAX önemsiz test aracı" seçeneğini tıklatın.

Her şey doğru şekilde çalışıyorsa, pencere arka planı yeşil olarak değişmelidir. Kurulumunuzda bir sorun varsa, pencere arka planı kırmızı olarak değişmeli ve hata bilgileri görüntülenecektir.

Aşağıdaki şekil, Visual Basic örneğinin merkezi kısmını göstermektedir.

```
Option Explicit

Private Sub Form_Click()

'*****************************************************************************
'* This simple example illustrates how to put and get a WebSphere MQ message to
'* and from a WebSphere MQ message queue. The data from the message returned by the
'*get is read and compared with that from the original message.
'*****************************************************************************
Dim MQSess As MQSession                 '* session object
Dim QMgr As MQQueueManager              '* queue manager object
Dim Queue As MQQueue                    '* queue object
Dim PutMsg As MQMessage                 '* message object for put
Dim GetMsg As MQMessage                 '* message object for get
Dim PutOptions As MQPutMessageOptions   '* get message option

Dim GetOptions As MQGetMessageOptions   '* put message options
Dim PutMsgStr As String                 '* put message data string
Dim GetMsgStr As String                 '* get message data string
'*****************************************************************************
'* Handle errors
'*****************************************************************************
On Error GoTo HandleError

'*****************************************************************************
```

```
'* Initialize the current position for the form
'*******************************************************************************
CurrentX = 0
CurrentY = 0

'*******************************************************************************
'* Create the MQSession object and access the MQQueueManager and (local) MQQueue
'*******************************************************************************
Set MQSess = New MQSession
Set QMgr = MQSess.AccessQueueManager("")
Set Queue = QMgr.AccessQueue("SYSTEM.DEFAULT.LOCAL.QUEUE", _
                      MQOO_OUTPUT Or MQOO_INPUT_AS_Q_DEF)

'*******************************************************************************
'* Create a new MQMessage object for use with put, add some data then create an
'* MQPutMessageOptions object and put the message
'*******************************************************************************
Set PutMsg = MQSess.AccessMessage()
PutMsgStr = "12345678 " & Time
PutMsg.MessageData = PutMsgStr
Set PutOptions = MQSess.AccessPutMessageOptions()
Queue.Put PutMsg, PutOptions

'*******************************************************************************
'* Create a new MQMessage object for use with get, set the MessageId (to that of
'* the message that was put), create an MQGetMessageOptions object and get the
'* message.
'*
'* Note: Setting the MessageId ensures that the get returns the MQMessage
'* that was put earlier.
'*******************************************************************************

Set GetMsg = MQSess.AccessMessage()
GetMsg.MessageId = PutMsg.MessageId
Set GetOptions = MQSess.AccessGetMessageOptions()
Queue.Get GetMsg, GetOptions
'*******************************************************************************
'* Read the data from the message returned by the get, compare it with
'* that from the original message and output a suitable message.
'*******************************************************************************
GetMsgStr = GetMsg.MessageData
Cls
If GetMsgStr = PutMsgStr Then
    BackColor = RGB(127, 255, 127)  '* set to green for ok
    Print
    Print "Message data comparison was successful."
    Print "Message data: """ & GetMsgStr & """"

Else
    BackColor = RGB(255, 255, 127)  '* set to amber for compare error
    Print "Compare error: "
    Print "The message data returned by the get did not match the " &_
    "input data from the original message that was put."
    Print
    Print "Input message data:        """ & PutMsgStr & """"
    Print "Returned message data: """ & GetMsgStr & """"
End If

Exit Sub
'*******************************************************************************
'* Handle errors
'*******************************************************************************
HandleError:
Dim ErrMsg As String
Dim StrPos As Integer

Cls
BackColor = RGB(255, 0, 0)  '* set to red for error
Print "An error occurred as follows:"
Print ""
If MQSess.CompletionCode <> MQCC_OK Then
    ErrMsg = Err.Description
    StrPos = InStr(ErrMsg, " ")            '* search for first blank
    If StrPos > 0 Then
        Print Left(ErrMsg, StrPos)         '* print offending MQAX object name
    Else
        Print Error(Err)                   '* print complete error object
    End If
    Print ""
    Print "WebSphere MQ Completion Code = " & MQSess.CompletionCode
    Print "WebSphere MQ Reason Code = " & MQSess.ReasonCode
    Print "(" & MQSess.ReasonName & ")"
```

```
Else
    Print "Visual Basic error:  " & Err
    Print Error(Err)
End If

Exit Sub

End Sub
```

MQAXCLSS örneğinin başlatılması

Bu örnek, kuyruk yöneticileri ve kuyruk nesneleri özelliklerine ve yöntemlerine göz atmanızı sağlar.

1. Kuyruk yöneticisini başlatın.
2. Open the file, MQAXCLSS.VBP, by double clicking on the document icon in Pencereler Explorer or by clicking File - Open from the file menu in Visual Basic.
3. Örneği başlatın.
4. Uygun kuyruk yöneticisini ve kuyruk adlarını girin ve ilgili düğmeleri tıklatın.

MQAXDLST örneği

Visual Basic MQAXDLST Sample, bir dağıtım listesinin, aynı iletiyi bir put ile iki kuyruğa göndermek için kullanılmasını gösterir. Örneği çalıştırmak için, MQAXCLSS örneğiyle aynı işlemi gerçekleştirin.

MQAX Starter sample for Microsoft Excel 95 or later

This section explains how to run the MQAX starter sample for Microsoft Excel 95 or later, MQAXTRIV.XLS.

MQAXTRIV.XLS örneği

1. Kuyruk yöneticisini başlatın.
2. Explorer 'da ya da File Manager'da, MQAX örneği MQAXTRIV.XLS.
3. Elektronik sayfadaki düğmeyi tıklatın.
4. Ekran, bir başarı (ya da başarısızlık) iletisiyle güncellenir.

Running the Bank demonstration with MQAX.XLS

Banka gösterisini yürütmek için bu adımları izleyin.

1. Kuyruk yöneticisini başlatın.
2. IBM WebSphere MQ MQSC komut dosyasını (BANK.TST. Bu, gerekli IBM WebSphere MQ kuyruk tanımlamalarını ayarlar.

   Bir MQSC komut dosyasını nasıl kullanacağını öğrenmek için Script (MQSC) Commandsbelgesine bakın.
3. MQAXBSRV.VBPkomutunu çalıştırın. Bu örnek program, bir arka uç uygulamasının benzetimini yapan sunucudur ve bu uygulamanın Microsoft Excel ile çalışması gerekir.
4. MQAX.XLS. Bu örnek, istemci IBM WebSphere MQ gösterimidir.
5. Listeden bir müşteri seçin.
6. **Gönder** düğmesini tıklatın.

Kısa bir duraksama (yaklaşık 3 saniye) sonra, değerler ile doldurulan alanlar ve bir çubuk grafiği görüntülenir.

ActiveX uyumlu bir WWW tarayıcısını kullanarak başlangıç örneği

Not: Bu örneği çalıştırmak için, ActiveX uyumlu bir Web tarayıcısı çalıştırıyor olmanız gerekir. Microsoft Internet Explorer (ancak Netscape Navigator' ı değil) uyumlu bir Web tarayıcısıdır.

HTML örneğini çalıştırma

Bu örnek, MQAX 'i hem VBScript hem de JavaScriptiçinden nasıl çağırabileceğiniz gösterilir.

1. Kuyruk yöneticisini başlatın.
2. Open the file, "MQAXTRIV.HTM", in your ActiveX compatible Web browser.

   Bunu, Windows Gezgini 'nde dosya simgesini çift tıklatarak ya da ActiveX uyumlu Web tarayıcınızın Dosya menüsünden Dosya-Aç seçeneğini belirleyebilirsiniz.
3. Ekrana gelen yönergeleri izleyin.

Özel notlar

Bu belge, ABD'de kullanıma sunulan ürünler ve hizmetler için hazırlanmıştır.

IBM, bu belgede sözü edilen ürün, hizmet ya da özellikleri diğer ülkelerde kullanıma sunmayabilir. Bulunduğunuz yerde kullanıma sunulan ürün ve hizmetleri yerel IBM müşteri temsilcisinden ya da çözüm ortağından öğrenebilirsiniz. Bir IBM ürün, program ya da hizmetine gönderme yapılması, açık ya da örtük olarak, yalnızca o IBM ürünü, programı ya da hizmetinin kullanılabileceğini göstermez. Aynı işlevi gören ve IBM'in fikri mülkiyet haklarına zarar vermeyen herhangi bir ürün, program ya da hizmet de kullanılabilir. Ancak, IBM dışı ürün, program ya da hizmetlerle gerçekleştirilen işlemlerin değerlendirilmesi ve doğrulanması kullanıcının sorumluluğundadır.

IBM'in, bu belgedeki konularla ilgili patentleri ya da patent başvuruları olabilir. Bu belgenin size verilmiş olması, patentlerin izinsiz kullanım hakkının da verildiği anlamına gelmez. Lisansla ilgili sorularınızı aşağıdaki adrese yazabilirsiniz:

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

Çift byte (DBCS) bilgilerle ilgili lisans soruları için, ülkenizdeki IBM'in Fikri Haklar (Intellectual Property) bölümüyle bağlantı kurun ya da sorularınızı aşağıda adrese yazın:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japonya

Aşağıdaki paragraf, İngiltere ya da bu tür hükümlerin yerel yasalarla uyuşmadığı diğer ülkelerde geçerli değildir: INTERNATIONAL BUSINESS MACHINES CORPORATION BU YAYINI, HAK İHLALİ YAPILMAYACAĞINA DAİR GARANTİLERLE TİCARİLİK VEYA BELİRLİ BİR AMACA UYGUNLUK İÇİN ZIMNİ GARANTİLER DE DAHİL OLMAK VE FAKS BUNLARLA SINIRLI OLMAMAK ÜZERE AÇIK YA DA ZIMNİ HİÇBİR GARANTİ VERMEKSİZİN "OLDUĞU GİBİ" ESASIYLA SAĞLAMAKTADIR. Bazı ülkeler bazı işlemlerde garantinin açık ya da örtük olarak reddedilmesine izin vermez; dolayısıyla, bu bildirim sizin için geçerli olmayabilir.

Bu yayın teknik yanlışlar ya da yazım hataları içerebilir. Buradaki bilgiler üzerinde düzenli olarak değişiklik yapılmaktadır; söz konusu değişiklikler sonraki basımlara yansıtılacaktır. IBM, önceden bildirimde bulunmaksızın, bu yayında açıklanan ürünler ve/ya da programlar üzerinde iyileştirmeler ve/ya da değişiklikler yapabilir.

Bu belgede IBM dışı Web sitelerine yapılan göndermeler kullanıcıya kolaylık sağlamak içindir ve bu Web sitelerinin onaylanması anlamına gelmez. Bu Web sitelerinin içerdiği malzeme, bu IBM ürününe ilişkin malzemenin bir parçası değildir ve bu tür Web sitelerinin kullanılmasının sorumluluğu size aittir.

IBM'e bilgi ilettiğinizde, IBM bu bilgileri size karşı hiçbir yükümlülük almaksızın uygun gördüğü yöntemlerle kullanabilir ya da dağıtabilir.

(i) Bağımsız olarak yaratılan programlarla, bu program da içinde olmak üzere diğer programlar arasında bilgi değiş tokuşuna ve (ii) değiş tokuş edilen bilginin karşılıklı kullanımına olanak sağlamak amacıyla bu program hakkında bilgi sahibi olmak isteyen lisans sahipleri şu adrese yazabilirler:

IBM Corporation
Yazılım Birlikte Çalışabilirlik Koordinatörü, Bölüm 49XA
3605 Highway 52 N

Rochester, MN 55901
U.S.A.

Bu tür bilgiler, ilgili kayıt ve koşullar altında ve bazı durumlarda bedelli olarak edinilebilir.

Bu belgede açıklanan lisanslı program ve bu programla birlikte kullanılabilecek tüm lisanslı malzeme, IBM tarafından, IBM Müşteri Sözleşmesi, IBM Uluslararası Program Lisansı Sözleşmesi ya da eşdeğer herhangi bir sözleşmenin kayıt ve koşulları altında sağlanır.

Burada belirtilen performans verileri denetimli bir ortamda elde edilmiştir. Bu nedenle, başka işletim ortamlarında çok farklı sonuçlar alınabilir. Bazı ölçümler geliştirilme düzeyindeki sistemlerde yapılmıştır ve bu ölçümlerin genel kullanıma sunulan sistemlerde de aynı olacağı garanti edilemez. Ayrıca, bazı sonuçlar öngörü yöntemiyle elde edilmiş olabilir. Dolayısıyla, gerçek sonuçlar farklı olabilir. Bu belgenin kullanıcıları, kendi ortamları için geçerli verileri kendileri doğrulamalıdırlar.

IBM dışı ürünlerle ilgili bilgiler, bu ürünleri sağlayan firmalardan, bu firmaların yayın ve belgelerinden ve genel kullanıma açık diğer kaynaklardan alınmıştır. IBM bu ürünleri sınamamıştır ve IBM dışı ürünlerle ilgili performans doğruluğu, uyumluluk gibi iddiaları doğrulayamaz. IBM dışı ürünlerin yeteneklerine ilişkin sorular, bu ürünleri sağlayan firmalara yöneltilmelidir.

IBM'in gelecekteki yönelim ve kararlarına ilişkin tüm bildirimler değişebilir ve herhangi bir duyuruda bulunulmadan bunlardan vazgeçilebilir; bu yönelim ve kararlar yalnızca amaç ve hedefleri gösterir.

Bu belge, günlük iş ortamında kullanılan veri ve raporlara ilişkin örnekler içerir. Örneklerin olabildiğince açıklayıcı olması amacıyla kişi, şirket, marka ve ürün adları belirtilmiş olabilir. Bu adların tümü gerçek dışıdır ve gerçek iş ortamında kullanılan ad ve adreslerle olabilecek herhangi bir benzerlik tümüyle rastlantıdır.

YAYIN HAKKI LİSANSI:

Bu belge, çeşitli işletim platformlarında programlama tekniklerini gösteren, kaynak dilde yazılmış örnek uygulama programları içerir. Bu örnek programları, IBM'e herhangi bir ödemede bulunmadan, örnek programların yazıldığı işletim altyapısına ilişkin uygulama programlama arabirimiyle uyumlu uygulama programlarının geliştirilmesi, kullanılması, pazarlanması ya da dağıtılması amacıyla herhangi bir biçimde kopyalayabilir, değiştirebilir ve dağıtabilirsiniz. Bu örnekler her koşul altında tüm ayrıntılarıyla sınanmamıştır. Dolayısıyla, IBM bu programların güvenilirliği, bakım yapılabilirliği ya da işlevleri konusunda açık ya da örtük güvence veremez.

Bu bilgileri elektronik kopya olarak görüntülediyseniz, fotoğraflar ve renkli resimler görünmeyebilir.

Programlama arabirimi bilgileri

Programlama arabirimi bilgileri (sağlandıysa), bu programla birlikte kullanılmak üzere uygulama yazılımları yaratmanıza yardımcı olmak üzere hazırlanmıştır.

Bu kitap, müşterinin IBM WebSphere MQhizmetlerini edinmek üzere program yazmasına olanak tanıyan, amaçlanan programlama arabirimlerine ilişkin bilgiler içerir.

Ancak, bu bilgiler tanılama, değiştirme ve ayarlama bilgilerini de içerebilir. Tanılama, değiştirme ve ayarlama bilgileri, uygulama yazılımlarınızda hata ayıklamanıza yardımcı olur.

Önemli: Bu tanılama, değiştirme ve ayarlama bilgilerini bir programlama arabirimi olarak kullanmayın; bu, değişiklik söz konusu olduğunda kullanılır.

Ticari Markalar

IBM, IBM logosu, ibm.com, IBM Corporation 'ın dünya çapında birçok farklı hukuk düzeninde kayıtlı bulunan ticari markalarıdır. IBM ticari markalarının güncel bir listesini Web üzerinde "Telif hakkı ve ticari marka bilgileri"www.ibm.com/legal/copytrade.shtmladresinde bulabilirsiniz. Diğer ürün ve hizmet adları IBM'in veya diğer şirketlerin ticari markaları olabilir.

Microsoft ve Windows, Microsoft Corporation'ın ABD ve/veya diğer ülkelerdeki ticari markalarıdır.

UNIX, The Open Group şirketinin ABD ve diğer ülkelerdeki tescilli ticari markasıdır.

Linux, Linus Torvalds'ın ABD ve/ya da diğer ülkelerdeki tescilli ticari markasıdır.

Bu ürün, Eclipse Project (http://www.eclipse.org/) tarafından geliştirilen yazılımları içerir.

Java ve Java tabanlı tüm markalar ve logolar, Oracle firmasının ve/ya da iştiraklerinin markaları ya da tescilli markalarıdır.

Parça numarası:

(1P) P/N: